National Library
of Canada

Bibliothèque nationale
du Canada

Canadian Theses Service

Service des thèses canadiennes

Ottawa, Canada
K1A 0N4

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

NL-339 (r. 88/04) c

Canada

UNIVERSITY OF ALBERTA

Constrained Long Range Predictive Control

BY

Rajendra Kumar Mutha ©

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

IN

Process Control

Department of Chemical Engineering

EDMONTON, ALBERTA

Fall, 1990

National Library of Canada
Bibliothèque nationale du Canada

Canadian Theses Service
Service des thèses canadiennes

Ottawa, Canada
K1A 0N4

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-315-64958-5

UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR:	Rajendra Kumar Mutha
TITLE OF THESIS:	Constrained Long Range Predictive Control
DEGREE:	Master of Science
YEAR THIS DEGREE GRANTED:	1990

Permission is hereby granted to UNIVERSITY OF ALBERTA LIBRARY to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific purposes only.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.

(SIGNED) M. Rajendra Kumar

C/O Suresh Kumar & Company
Kajee Street
Rajahmundry
India–533 101

DATED: [illegible] 19 90

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled *Constrained Long Range Predictive Control* submitted by *Rajendra Kumar Mutha* in partial fulfilment of the requirements for the degree of *Master of Science* in *Process Control*.

Dr. S. L. Shah (Supervisor)

Dr. R. K. Wood

Dr. K. Bollinger

Date July 25. 19.92

to my parents

ABSTRACT

This thesis is concerned with the practical implementation of one member of constrained long range range predictive control algorithm, namely the generalized predictive control (GPC of Clarke *et al.* [13,14]). It evaluates the performance of adaptive constrained GPC on a single-input, single-output (SISO) system and a fixed parameter or non-adaptive constrained GPC as applied to multi-input multi-output (MIMO) systems, through simulations and experimental runs. Special emphasis is placed on illustrating the influence of constraints on the performance of the control algorithm. Analytical solutions of constrained GPC algorithm are also presented for some specific values of controller tuning and are of practical importance.

The simulations and experimental runs demonstrate the effectiveness of the constrained GPC algorithm. The experimental runs evaluate the performance of adaptive constrained SISO GPC on processes with non-linearities, variable time delays and time varying dynamics. The MIMO GPC is implemented with a fixed parameter model. The simulations and experimental runs on MIMO processes illustrate the performance of GPC in the presence of process non-linearities, measurement noise, etc. It also shows the effect of variation of various tuning parameters and the effect of load disturbances with and without feed-forward compensation. Since multivariable GPC is implemented with a fixed parameter model, the study also shows the effect of model plant steady state gains mismatch on the performance of the system and thereby suggests a guideline for selecting various tuning parameters that can effectively handle such mismatch.

This thesis also presents an analysis of the effects of scaling the manipulated and the controlled variables of a MIMO system under long range predictive control algorithm.

It describes the necessary and sufficient conditions for optimally scaling a square matrix in l_2-norm. The study demonstrates the increase in numerical robustness of an system with optimally scaled inputs. The optimal scaling methods for systems with equal number of inputs and outputs is extended to systems which have an unequal number of inputs and outputs.

ACKNOWLEDGEMENTS

I would like to take this opportunity to offer my sincere thanks to Dr. S.L. Shah for his supervision, guidance and proof-**re**(a)bf ding of this thesis; Dr. D.G. Fisher for his guidance and direction.

I would also like to thank my colleagues at the University of Alberta who have made the last two years possible, productive and enjoyable; David Shook for his constant help and piling up things all over the work place; Coorous Mohtadi for his direction and 'Ghetto Blaster'; Kun-Yu Kwok, Eric Lau, Fengxi Zhou and Weiping Lu for their help and suggestions; my 'lunch-buddies' Greg Holloway and Ravindra Yaparpalvi for bearing up with me; Don Sutherland, Walter Bondez and Keith Faulder of the electronic, instrument and machine shops for their help in modifying and maintaining the stirred tank heater; my brother Kailash Mutha for his suggestions and in assisting me master 'C'. Many thanks are also due to Mary Bourke (for her Feline support¿!#), Roman Walther and Niu Shaohoa.

Financial support from Dow Chemical Canada Ltd. and Imperial Oil Ltd. is gratefully acknowledged.

I would also take this opportunity in thanking my parents for their constant support and encouragement over so many years.

Contents

1 Introduction **1**
- 1.1 Adaptive or Self-Tuning Control 2
- 1.2 Long Range Predictive Control 3
- 1.3 Objective of Study 5
- 1.4 Structure of the Thesis 6

2 Constrained Generalized Predictive Control **8**
- 2.1 Introduction 8
- 2.2 The Basic SISO GPC 9
  - 2.2.1 The CARIMA Process Model 9
  - 2.2.2 Long Range Prediction 10
  - 2.2.3 The Predictive Control Law 12
  - 2.2.4 The Control Horizon 14
- 2.3 Basic Multivariable GPC 15
  - 2.3.1 The MV CARIMA Process Model 15
  - 2.3.2 Long Range Prediction in MGPC 16
  - 2.3.3 The Multivariable Predictive Control Law 16
- 2.4 Extensions to the Basic GPC Algorithm 18
  - 2.4.1 T(q^{-1}): Load Disturbance Rejection Polynomial 18
  - 2.4.2 Feedforward Compensation 20
  - 2.4.3 Constraints 21
  - 2.4.4 Output Weighting 23
- 2.5 Long Range Predictive Identification Algorithm 24

3 Analytical Solution for Simple Cases of Constrained GPC **26**
- 3.1 Development of Analytical Solution 27
  - 3.1.1 Introduction 27
  - 3.1.2 Constrained SISO GPC for $NU = 1$ 29
  - 3.1.3 Constrained SISO GPC for $NU = 2$ 30
  - 3.1.4 Simulations 34
  - 3.1.5 Constrained MIMO GPC for $NU = 1$ 37
- 3.2 Geometric Interpretation of Constrained GPC 43
  - 3.2.1 Geometric Shape of the GPC Cost Function 43

3.2.2 Geometric Shapes of the Feasible Region . . . 44
3.2.3 Geometric Interpretation of the Optimal Solution . . . 47
3.3 Conclusions . . . 50

4 Performance Evaluation of the MGPC **51**
4.1 Software Development for MGPC . . . 51
4.1.1 The MGPC Algorithm . . . 55
4.2 The Shell Control Problem–A Case Study . . . 58
4.2.1 The Shell Control Problem . . . 58
4.2.2 Interaction Between Different Outputs of the Closed Loop System . . . 64
4.2.3 Load Rejection, with and without Feedforward Compensation . . . 67
4.2.4 Effect of Model Process Mismatch . . . 70
4.3 Conclusions . . . 88

5 Experimental Application of GPC **89**
5.1 Implementation Details . . . 89
5.2 Control of Stirred Tank Heater with Adaptive SISO GPC . . . 92
5.2.1 Process Equipment . . . 92
5.2.2 SISO Experimental Runs . . . 94
5.3 Control of Stirred Tank with Fixed Parameter MGPC . . . 102
5.3.1 Process Equipment . . . 102
5.3.2 Simulation of the MIMO Stirred-tank Process . . . 111
5.3.3 MIMO Experimental Runs on Stirred Tank . . . 112
5.4 Arterial Pressure Control with Adaptive SISO GPC . . . 130
5.4.1 Process Description . . . 131
5.4.2 Experimental Results . . . 133
5.5 Conclusions . . . 136

6 Scaling in Multivariable Systems **138**
6.1 Introduction . . . 139
6.2 Best l_2 Scaling of a Matrix . . . 143
6.2.1 SVD—A Physical Interpretation . . . 143
6.2.2 Condition for Best Scaling of a Matrix . . . 144
6.3 Use of Scaling In MIMO Systens . . . 148
6.3.1 Formulation of Scaled MGPC . . . 149
6.3.2 Scaling And Their Effects . . . 151
6.3.3 Simulation Example . . . 152
6.3.4 Discussion . . . 157
6.4 Scaling for Systems with an Unequal Number of Inputs and Outputs . . . 159
6.4.1 Selection of the Submatrix . . . 160
6.4.2 Example . . . 161
6.5 Conclusions . . . 162

7 Conclusions **164**
7.1 Recommendations for Future Work 167

References **168**

Appendix **172**

A Analytical Solution of Some Cases of Constrained GPC **172**
A.1 Analytical Solution of SISO GPC for $NU = 2$ 172
A.2 Analytical Solution of MIMO (2×2) GPC for $NU = 1$ 176

B Proof of equivalence of unscaled and D_2 scaled MGPC algorithm **179**

C Listing of the Software for Implementing MGPC **182**

List of Tables

3.1 Condition on Kuhn-Tucker multipliers for different formulations of the problem . . . 29

4.1 The process model of transfer function parameters . . . 62
4.2 The suggested model process gains mismatch in the Shell control problem . 62

5.1 Nominal operating conditions for SISO configuration of stirred tank heater 92
5.2 Nominal operating conditions for MIMO configuration of stirred tank heater 103

List of Figures

1.1 Block diagram layout of an adaptive controller 2

2.1 Configuration of LRPI and GPC in the Closed Loop 25

3.1 Simulation of a distillation column under unconstrained GPC 35
3.2 Simulation of a distillation column under rate constrained GPC 36
3.3 Simulation of a distillation column under amplitude constrained GPC . . . 38
3.4 Simulation of a distillation column under rate plus amplitude constrained GPC 39
3.5 Feasible region of rate constraints 45
3.6 Feasible region of the amplitude constraints 46
3.7 Feasible region of the rate plus amplitude constraints 48
3.8 Optimal solution of a constrained cost function 49

4.1 Schematic diagram of the heavy oil fractionator used as the Shell control problem 59
4.2 Open loop step response characteristics of the Shell control problem 63
4.3 Interaction between different outputs of the closed loop system in the Shell control problem with rate constraints of ±0.2 65
4.4 Interaction between different outputs of the closed loop system in the Shell control problem with rate constraints of ±0.0025 66
4.5 Load rejection by MGPC without feedforward compensation for the Shell control problem with $NU = 1$ 68
4.6 Load rejection by MGPC without feedforward compensation for the Shell control problem with $NU = 2$ 69
4.7 Load rejection by MGPC with feedforward compensation for the Shell control problem with $NU = 1$ 71
4.8 Load rejection by MGPC with feedforward compensation for the Shell control problem with $NU = 2$ 72
4.9 Load rejection by MGPC with feedforward compensation for the Shell control problem with $NU = 3$ 73
4.10 Control of Shell control problem with MPM in G_{11} 75
4.11 Control of Shell control problem with 'default' control setting for MPM in G_{11}, G_{22} and G_{33} 77

4.12 Control of Shell control problem with Λ weighting for MPM in G_{11}, G_{22} and G_{33} . . . 78
4.13 Control of Shell control problem with Λ and y_{wt} weighting for MPM in G_{11}, G_{22} and G_{33} . . . 79
4.14 Control of Shell control problem with y_{wt} weighting for MPM in G_{11}, G_{22} and G_{33} . . . 81
4.15 Control of Shell control problem with relaxed rate constraints for MPM in G_{11}, G_{22} and G_{33} . . . 82
4.16 Control of Shell control problem rate constraints for MPM in G_{11}, G_{21} and G_{31} . . . 84
4.17 Control of Shell control problem with second order $T(q^{-1})$ for MPM in G_{11}, G_{21} and G_{31} . . . 85

5.1 Schematic diagram of Continuously Stirred Tank Heater for SISO configuration . . . 93
5.2 The effect of rate constraints on the control of the stirred tank heater in SISO configuration . . . 97
5.3 Performance of an unconstrained SISO GPC on the stirred tank heater . . . 98
5.4 Performance of a rate constrained SISO GPC on the stirred tank heater . . . 99
5.5 Performance of a output constrained SISO GPC on the stirred tank heater 101
5.6 Schematic diagram of stirred tank heater for MIMO configuration . . . 104
5.7 Open loop response of the temperature (^{0}C) as output (y_1) with percentage cold water valve opening (u_1) as input for the tank in MIMO configuration 105
5.8 Open loop response of the temperature (^{0}C) as output (y_1) with percentage hot water valve opening as input (u_2) for the tank in MIMO configuration . 105
5.9 Open loop response of the pecentage tank level (y_2) as output with percent cold water valve opening (u_1) as input for the tank in MIMO configuration . 106
5.10 Open loop response of percentage tank level (y_2) as output with percentage hot water valve opening (u_2) as input for the tank in MIMO configuration . 106
5.11 Open loop response of temperature (^{0}C) as output (y_1) at thermocouple # 2 as output with percentage cold water valve opening (u_1) as input for the tank in MIMO configuration . . . 108
5.12 Open loop response of temperature (^{0}C) as output (y_1) at thermocouple # 2 as output with percentage hot water valve opening (u_2) as input for the tank in MIMO configuration . . . 108
5.13 The non linear characteristics of the MIMO experimental equipment . . . 110
5.14 Simulation of the 2 × 2 stirred tank process for different output prediction horizons . . . 113
5.15 Simulation of the 2 × 2 stirred tank process for evaluating the effect of output weighting and rate constraints . . . 114
5.16 Control of stirred tank with a well tuned unconstrained MGPC . . . 116
5.17 Control of stirred tank with a well tuned constrained MGPC . . . 117
5.18 Effect of constraints on the control of stirred tank with MGPC . . . 119

5.19 Effect of variation in maximum prediction horizon in MGPC on the control of stirred tank 121
5.20 Effect of variation in output weighting in MGPC on the control of stirred tank 123
5.21 Control Of stirred tank for temperature measured at thermocouple # 2 with MGPC 125
5.22 Effect of variation in control weighting in MGPC on the control of stirred tank 127
5.23 Control of stirred tank heater in MIMO configuration with tuned PID Controllers 129
5.24 Block diagram of the experimental equipment for adaptive blood pressure control 132
5.25 Arterial pressure control of a dog with unconstrained GPC 134
5.26 Arterial pressure control of a dog with rate constrained GPC 135

6.1 Simulation of an ill-conditioned system G(S) with no scaling 141
6.2 Block diagram representation of the scaled system 142
6.3 Simulation of an ill-conditioned system with system outputs scaled 154
6.4 Simulation of an ill-conditioned system with system inputs scaled 155
6.5 Simulation of an ill-conditioned system with both system inputs and outputs scaled 156

Chapter 1

Introduction

Automatic control of industrial chemical processes is a challenging task due to the presence of large time delays, non-linearities, complex process dynamics, interaction from other control loops, corruption of measurements by noise, unmeasurable load disturbances, actuator non-linearities such as saturation, etc. Control of a process with conventional control schemes such as multiloop PID schemes is often not satisfactory, depending on the severity of the above mentioned characteristics. The fixed parameter PID control algorithm, though widely used in industry, suffers from many drawbacks:

- Time delays can only be handled by detuning the loop. This leads to a relatively sluggish response from the process.
- Physical constraints of a process cannot be incorporated in the control algorithm.
- Tuning of a PID loop is time consuming. The tuning of a multiloop PID controller is at best a non-trivial task. In the absence of a model, it is performed on-line by an iterative experimental approach, *i.e.* 'trial and error'. The development of a mathematical model accounting for all the non-linearities of a process is not easy. The linear model of a process can help in selecting the initial tuning constants but is not good enough to give the 'best' possible performance over the whole range of operation.

- Change of process dynamics (due to catalyst decay, fouling of heat exchangers, change in production levels, variation in raw material quality and quantity etc.) will lead to a degraded performance unless the controller is retuned.

The above problems can be overcome by the use of a control strategy first proposed by Kalman [23].

1.1 Adaptive or Self-Tuning Control

Adaptive controllers[1] are capable of identifying the parameters of the process on-line and are able to compensate for the variations in the characteristics of a process. The general structure of this control strategy is shown in figure 1.1.

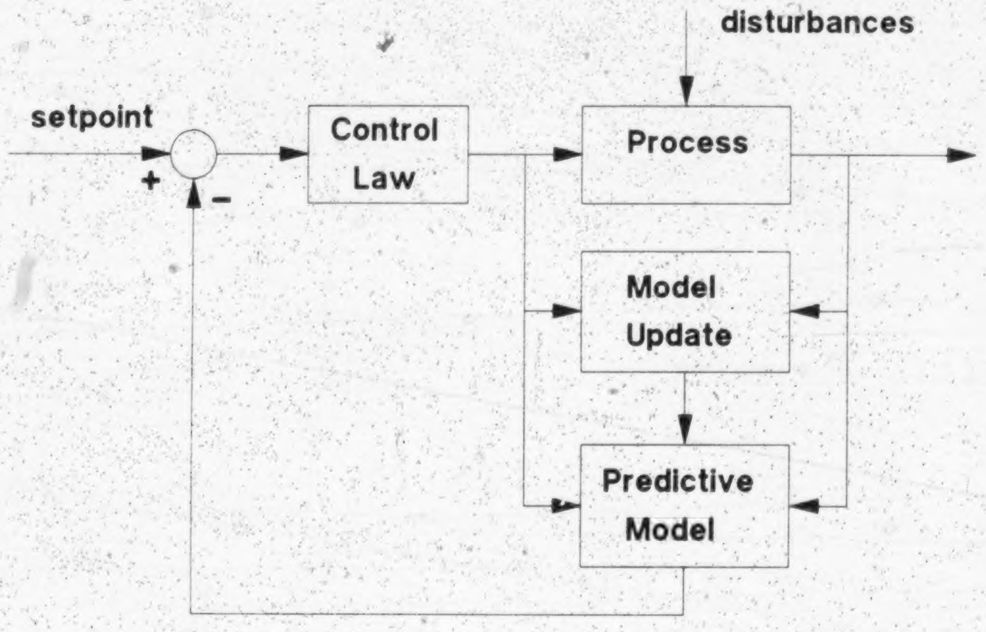


Figure 1.1: Block diagram layout of an adaptive controller

---

[1] the terms adaptive control and self-tuning control are considered synonymous and used interchangeably in this thesis

One of the major benefits from the use of self-tuning controllers is the potential increase in the speed of controller commissioning. Furthermore, the use of self-tuners would result in 'expert tuning by non-expert personnel', *i.e.* widespread use of self-tuners can increase the overall quality of control throughout the industry by making the quality of control in individual loops less dependent upon the expertise of control engineers.

1.2 Long Range Predictive Control

Large time delays are the main source of difficulty for PID controllers (Stephanopoulos [54]). The problem of time delay was solved with the development of predictive control strategies; where control is not based on current output but on the predicted output some time in the future. The prediction is based on knowledge of past control actions and the process model. In the late 1950's time delay compensation was provided for fixed parameter conventional controllers by the Smith predictor algorithm (Smith [52]). This was subsequently followed by the minimum variance controller of Åström[2] and generalized minimum variance (GMV) controller of Clarke and Gawthrop [9,10]. Both these controllers are self-tuning and based on k-step ahead prediction of the output, where k is the assumed time delay of the process. Though these algorithms gained widespread popularity, they suffer from two specific problems: if the assumed delay is incorrect or time varying, or if the process is a non-minimum phase system, the controller must be detuned to preserve stability.

Two alternative control algorithms are capable of overcoming the above problems; *pole-placement* and *long range prediction*. Pole-placement (Wellstead *et al.* [58], Åström and Wittenmark [3]) is not based on a zero cancellation-type control strategy and hence in principle is capable of handling time delays and non-minimum phase systems.

The second alternative, long-range predictive control (LRPC), is an extension of the predictive control strategies and is based on a *range* of future predictions. Specifying a prediction range well beyond the largest expected pure time delay results in a de-tuned but *robust* control performance. This algorithm is also free of zero cancellation-type strategy and is relatively insensitive to time delays and non-minimum phase systems.

The early LRPC algorithms such as identification/command (IDCOM of Richalet *et al.* [44]) and dynamic matrix control (DMC of Cutler and Ramaker [15]) were based on deterministic impulse or step response models. Though easy to obtain, the relatively large parameters in the process model make these immediately not suitable for adaptive implementation.

LRPC algorithms based on transfer function or input-output models have the potential to become self-tuning as they are capable of representing the processes in a simple manner with fewer parameters. The important LRPC algorithms based on such models are: Peterka's infinite stage predictive controller [40], Multistep Multivariable Adaptive Regulator (MUSMAR) approach of Mosca [38], extended horizon adaptive control (EHAC) algorithm of Ydstie [59,60] based upon GMV with the output prediction well beyond the time delay, extended prediction self-adaptive control (EPSAC) algorithm of De Keyser *et al.* [16,17] which is a modification of GMV where the output prediction is extended over a horizon of future values, and the generalized predictive control (GPC) algorithm of Clarke *et al.* [13,14]. The latter is a generalization of previous multistep predictive algorithms as well as being the natural long-range extension of GMV.

Generalized predictive control involves explicit identification of the process and the minimization of a multistep quadratic cost function of future predicted errors to compute the incremental control signal. Use of controlled auto-regressive integrated moving average (CARIMA) model structure for the process model ensures inherent integral action (Clarke [12]). The calculation of the control signal by minimizing the quadratic cost function is based on the assumption that after a user specified *control horizon* all control increments are assumed to be zero. This idea is borrowed from the DMC algorithm of Cutler and Ramaker [15]. Proper selection of prediction horizon and the control horizon makes GPC relatively insensitive to unknown or variable time delay and non-minimum phase processes.

The physical constraints of a process, such as saturation of actuators, can be incorporated in the LRPC algorithms. For GPC, the physical constraints can be translated as constraints on the incremental control signal and the quadratic cost function of the algorithm can be subjected to the constraints. This will ensure that the control algorithm always generates physically realizable signal (if they exists).

1.3 Objective of Study

The LRPC algorithm, GPC, was experimentally evaluated on single-input single-output (SISO) and multi-input multi-output (MIMO) processes. Performance of an *adaptive constrained SISO GPC* was experimentally evaluated on two processes. Performance of a *fixed parameter or non-adaptive multivariable GPC* was evaluated through experiments and simulations. The objectives of this evaluation were:

- to develop software for implementing multivariable GPC capable of controlling processes with n outputs and m inputs, where n is less than or equal to or greater than m, with constraints on amplitude, incremental change of control signal, and amplitude constraints on the process outputs.
- to develop analytic solutions for special cases of $NU = 1$ and 2 for SISO and $NU = 1$ for multivariable GPC with constraints on rates and/or amplitudes of the manipulated variables.
- to evaluate the performance of adaptive constrained SISO GPC for processes with non-linearities, variable time delays, measurement noise, time varying dynamics, etc.
- to evaluate the performance of fixed parameter or non-adaptive multivariable GPC for processes with non-linearities, measurement noise, etc. Since this is a fixed parameter control algorithm, the study also shows the effect of model process mismatch in LRPC algorithms, by simulations and through experimental runs and thereby suggests guidelines for selecting certain tuning parameters in multivariable GPC.

- to compare the performance of constrained and unconstrained LRPC algorithms for SISO and MIMO processes.

1.4 Structure of the Thesis

This thesis is mainly concerned with the *practical* application of a constrained LRPC algorithm, GPC, on SISO and MIMO systems. The structure of the thesis is outlined below with emphasis on the contributions of this work. A more detailed introduction is given at the start of each individual chapter.

Chapter 2 reviews the basic GPC algorithm and existing extensions to incorporate various design polynomials. It also formulates GPC for handling constraints on the controlled and manipulated variables in the algorithm. This chapter also describes the extension of GPC to multivariable systems.

Chapter 3 develops an analytical solution for special cases of constrained GPC algorithm. The analytical solutions are developed based on the Kuhn-Tuckers multiplier method. This chapter presents a solution to the constrained SISO GPC for control horizon values of 1 and 2 and a solution to constrained multivariable GPC for a control horizon of 1. This chapter also gives a geometrical interpretation of the constraints and the cost function of GPC.

Chapter 4 evaluates the performance of GPC with the help of simulations. A case study of the Shell control problem [42] is presented in this chapter. The simulations show the effect of variation of various 'tuning knobs' of the fixed parameter multivariable control algorithm. The effect of constraints on the performance of the system is illustrated through some of the simulations in this chapter. A number of simulations are presented to show the performance of multivariable GPC algorithm in the presence of various degrees of model process mismatch in the gain. Based on these simulations, a suggestion on tuning of output weighting (y_{wt}) is suggested. The simulations also point out the limitations of fixed parameter multivariable GPC in controlling systems with very large model process mismatch.

Chapter 5 describes the application of adaptive constrained SISO GPC and fixed parameter constrained multivariable GPC to control the stirred tank. Special emphasis is placed on highlighting the benefits of using the constrained algorithm. The multivariable experiments also demonstrates that output weighting is a critical controller tuning parameter. A few simulations of the MIMO process used for experimental setup are also presented in this chapter for comparison with the experimental results. This chapter also describes a successful application of adaptive constrained SISO GPC to regulate the mean arterial blood pressure in a dog. This is a highly non-linear and time varying system and the successful application of GPC on this system demonstrates the effectiveness of the algorithm.

Chapter 6 analyzes the effects of scaling on the closed loop performance of a multivariable system. It describes the necessary and sufficient conditions required for a best scaled square matrix in l_2 norm. The effects of scaling controlled and/or manipulated variables are demonstrated on a MIMO system under GPC algorithm. The optimal scaling method for systems with an equal number of inputs and outputs is extended to systems with an unequal number of inputs and outputs.

Chapter 7 draws conclusions from the results presented in the thesis and provides suggestions for future work.

Chapter 2

Constrained Generalized Predictive Control

Generalized predictive control (GPC) of Clarke *et al.* [13,14] is a long-range, multi-step predictive control algorithm. Adaptive SISO GPC is experimentally evaluated on a continous stirred-tank heater as described in chapter 5. The multivariable extension of generalized predictive controller (MGPC) is described in Mohtadi [35] and Shah *et al.* [46]. A fixed-parameter or non-adaptive MGPC was used for simulations discussed in chapter 4 and experimental runs described in chapter 5. The development and interpretation of the control algorithm, with constraints on process inputs and outputs, forms the major subject of this chapter. The estimation scheme used for SISO systems is also briefly described.

2.1 Introduction

GPC is one of the more recent long range predictive control algorithms. It combines several important features:

1. The use of *long-range prediction* (Richalet *et al.* [44]) over a finite multi-step horizon (from N_1 to N_2 samples in the future).This is expected to increase the robustness of the control in the presence of unknown, or varying time-delays, as well as non-minimum phase plants.

2. The use of a CARIMA (Tuffs [56]) process model to provide these predictions in an incremental form. This inherently introduces integral action in the closed loop and

aids in achieving offset-free control for step-type or random-walk disturbances and set-point changes.

3. The assumption of a *control horizon*, NU (Cutler and Ramaker [15]), after which all projected control increments are assumed to be zero. By constraining the calculation for the control signal in this fashion, GPC reduces the computational effort significantly and simplifies control of non-minimum phase plants.

4. The optional weighting of control increments in the cost function ensures offset-free rejection of non-stationary disturbances as a result of inherent integral action.

This chapter is concerned with the development of basic SISO GPC and MGPC, with constraints on the controller outputs and process outputs, as well as a brief discussion of the identification algorithm. The chapter is organized as follows: sections 2.2 and 2.3 describe the development of basic GPC algorithm for SISO and MIMO systems respectively. The next section, 2.4, incorporates various design features and constraints into the basic GPC algorithm. The identification algorithm used in adaptive SISO GPC for conducting experimental runs is briefly described in section 2.5.

2.2 The Basic SISO GPC

The development of the SISO GPC proceeds initially by definition of an appropriate model, suitable for the prediction of future process outputs. The algorithm is extended to produce a range of future predictions. Subsequently it is formulated as a quadratic cost-function, which can be minimized to generate the required control signal.

2.2.1 The CARIMA Process Model

The controlled auto-regressive integrated moving average process model is defined as follows:

$$A(q^{-1})y(t) = B(q^{-1})u(t-1) + \frac{C(q^{-1})}{\Delta}\xi(t) \tag{2.1}$$

where $y(t)$ and $u(t-1)$ are process output and input respectively; Δ is the difference operator, $\Delta = 1 - q^{-1}$, $\xi(t)$ is an uncorrelated random noise sequence with zero mean and A and B are polynomials in the backward shift operator q^{-1} *i.e.*

$$A(q^{-1}) = 1 + a_1 q^{-1} + \ldots + a_{\delta A} q^{-\delta A}$$

$$B(q^{-1}) = b_0 + b_1 q^{-1} + \ldots + b_{\delta B} q^{-\delta B}$$

If a plant has a non-zero time delay, then the leading elements of the polynomial $B(q^{-1})$ are zero. Although $C(q^{-1})$ is another polynomial in the delay operator, it will be chosen as 1 for simplicity. Thus the CARIMA representation looks like:

$$A(q^{-1})y(t) = B(q^{-1})u(t-1) + \frac{\xi(t)}{\Delta} \tag{2.2}$$

There are many reasons for the choice of this type of model structure; the essentially simple linear difference equation is suitable for discrete computer implementation, and readily manipulated into a predictive form. The particular choice of the CARIMA model is made on the basis that the noise term $(C(q^{-1})\xi(t))/\Delta$ is a more realistic model of typical process load disturbances, admitting interpretation as either random steps at random intervals or as drifts. In practice, however, the main justification for the use of this model structure is that it forces the use of incremental data in the control calculations due to the presence of an integrator in the model.

Although the model uses linear parameters, it can be justified as a form of linearisation of the process about its operating point. It will be assumed throughout the remainder of this chapter that the parameters of this model are known.

2.2.2 Long Range Prediction

A long range predictive controller requires a series of predictions of future values of the output, *i.e.* $\{\hat{y}(t+j), j = 1, 2, \ldots\}$, given information up to time t and assumptions about future control activity. Thus, following the standard derivation of Clarke *et al.* [13], the

CARIMA plant model description of (2.1) is rearranged into a form suitable for the generation of j-step ahead predictions, using the Diophantine identity:

$$1 = E_j(q^{-1})A(q^{-1})\Delta + q^{-j}F_j(q^{-1}) \tag{2.3}$$

where E_j and F_j are polynomials of degree $j-1$ and $\delta A - 1$ respectively, and are uniquely defined for a given $A(q^{-1})$ and the prediction interval j. Substitution of this identity in 2.1 gives a prediction equation for the $y(t+j)$:

$$\hat{y}(t+j|t) = G_j(q^{-1})\Delta u(t+j-1) + F_j(q^{-1})y(t) \tag{2.4}$$

where $G_j(q^{-1}) = E_j(q^{-1})B_j(q^{-1})$ and the associated *prediction error* is,

$$e(t+j|t) = E_j(q^{-1})\xi(t+j) \tag{2.5}$$

Thus, in principle, it is possible to numerically solve the Diophantine equation 2.3 for each prediction interval, j, and to substitute it into 2.4 to obtain the required predictions. In practice it is far more efficient to solve 2.1 for initial values of E_i and F_i and then to calculate subsequent values of E_{i+1} and F_{i+1} using a recursion of the Diophantine equation (McIntosh [30]) .

Alternatively the prediction equation 2.4 can be rewritten as:

$$\begin{aligned} \hat{y}(t+j|t) &= \tilde{G}_j(q^{-1})\Delta u(t+j-1) + \bar{G}_j(q^{-1})\Delta u + F_j(q^{-1})y(t) && (2.6)\\ &= \tilde{G}_j(q^{-1})\Delta u(t+j-1) + f(t+j) && (2.7)\end{aligned}$$

Thus the prediction $\hat{y}(t+j)$ can be split into two distinct parts:

- $f(t+j)$, the prediction of the output assuming no future changes in the control signal *i.e.* $\Delta u(t+j-1) = 0, j = 1, 2 \ldots$. This corresponds to the ordinates of the process 'free-response'.

- $\tilde{G}_j(q^{-1})\Delta u(t+j-1)$, the prediction of the plant output that is dependent upon the future control increments yet to be calculated. This represents a convolution of the future control increments and the 'step-response' of the process and thus the individual $\tilde{G}$ parameters correspond to the appropriate ordinates of process step-response.

While implementing the GPC algorithm for simulations and experimental runs, the free-response of the process was calculated by setting $\Delta u(t+j-1) = 0, j = 1, 2 \ldots$, without using the Diophantine equation, because it is computationally less intensive for adaptive control.

2.2.3 The Predictive Control Law

The prediction equation (2.4) not only allows the calculation of future predictions given future control signals, but also allows the calculation of future control signals given the desired value of future predictions. For GPC this calculation proceeds by definition of a quadratic cost-function representing the difference between future predictions of the output and a prespecified set-point *trajectory* $\{w(t+j), j = 1, 2, \ldots\}$:

$$J_{GPC} = E\{\sum_{j=N_1}^{N_2}[y(t+j) - w(t+j)]^2 + \sum_{j=1}^{N_2}\lambda(j)[\Delta u(t+j-1)]^2\} \tag{2.8}$$

where

- N_1 is the *minimum prediction* horizon,
- N_2 is the *maximum prediction* horizon,
- $\lambda(j)$ is a *weighting* upon future control increments, and
- E is the expectation operator.

The above equation (2.8) minimizes the prediction error over a range of future predictions, starting from N_1 samples into the future and ending after N_2 samples. The future predictions are calculated using the equation (2.7). The minimum prediction horizon, N_1, is normally assumed to be 1; but for processes with known minimum time-delay, it can be chosen equal to or greater than minimum time-delay, to minimize computation for the same range of predictions. Also for non-minimum phase processes, it can be chosen greater than 1 for the controller to overlook the non-minimum phase behavior of the process. The maximum prediction horizon, N_2, in general, should be chosen to extend across most of the process response that is expected to be affected by any change in the current control signal. Thus, it should be chosen at least to be greater than the degree of the $B(q^{-1})$ polynomial of CARIMA process model (equation 2.2) and if possible should correspond to the rise-time of the plant itself.

The derivation of the control law from the cost-function in equation 2.8 is extensively described in the literature (Clarke *et al.* [13,14], Mohtadi [35]). It can be proved that the cost function (2.8) can be vectorized as:

$$J_{GPC} = [\mathbf{G}\tilde{\mathbf{u}} + \mathbf{f} - \mathbf{w}]^T[\mathbf{G}\tilde{\mathbf{u}} + \mathbf{f} - \mathbf{w}] + \lambda\tilde{\mathbf{u}}^T\tilde{\mathbf{u}} \tag{2.9}$$

and its solution is :

$$\tilde{\mathbf{u}} = [\mathbf{G}^T\mathbf{G} + \lambda I]^{-1}\mathbf{G}^T(\mathbf{w} - \mathbf{f}) \tag{2.10}$$

where

$$\begin{aligned}
\mathbf{w} &= [w(t+N_1)\ \ w(t+N_1+1)\ \ \ldots\ \ w(t+N_2)]^T \\
\mathbf{f} &= [f(t+N_1)\ \ f(t+N_1+1)\ \ \ldots\ \ f(t+N_2)]^T \\
\tilde{\mathbf{u}} &= [\Delta u(t)\ \ \Delta u(t+1)\ \ \ldots\ \ \Delta u(t+N_2-1)]^T
\end{aligned}$$

and $f(t+j)$ is the free response as described in the section(2.2.2) and $\mathbf{G}$ is a matrix of dimension $(N_2 - N_1 + 1) \times N_2$:

$$\mathbf{G} = \begin{bmatrix} g_{N_1-1} & \cdots & g_0 & 0 & 0 & \cdots & 0 \\ g_{N_1} & \cdots & g_1 & g_0 & 0 & \cdots & 0 \\ \vdots & \ddots & & & & & \vdots \\ g_{N_2-1} & \cdots & & & & & g_0 \end{bmatrix} \tag{2.11}$$

where each g_i is the value of the step response of the CARIMA process model at the i_{th} sampling interval. Note that if the plant has a physical dead time $d \geq N_1$, then the first

$d+1-N_1$ rows and last d columns of **G** will be entirely zeros. As GPC is a *receding-horizon* control policy, only the first element of the control sequence is actually implemented and, therefore, it is necessary to calculate only the first row of $[\mathbf{G}^{\mathbf{T}}\mathbf{G}+\lambda\mathbf{I}]^{-1}\mathbf{G}^{\mathbf{T}}$, at each sample interval.

2.2.4 The Control Horizon

While implementing the adaptive GPC on a process, it is necessary to invert on-line the matrix in equation 2.10. The dimension of this matrix is $N_2 \times N_2$ and results in heavy computational load, since N_2 is typically large. Moreover, if the plant has physical dead-times such that $d \geq N_1$, the matrix $\mathbf{G}^T\mathbf{G}$ is singular and a finite non-zero value of control weighting, λ, is necessary for $[\mathbf{G}^{\mathbf{T}}\mathbf{G}+\lambda\mathbf{I}]^{-1}$ to exist. These limitations can be overcome by the use of a specified control horizon.

The specification of a control horizon, NU, is an idea borrowed from the dynamic matrix control (DMC) algorithm of Cutler and Ramaker [15] . It assumes that after an interval of NU future samples, the projected control increments are zero, *i.e.*

$$\Delta u(t+j-1)=0 \quad ; \; j > NU \tag{2.12}$$

The control horizon, NU, represents the number of non-zero control increments the controller is free to select for minimizing the cost function given by equation (2.8). The incorporation of this idea into algorithm results in two immediate advantages:

- *Simplification of the control calculation:* Incorporation of the control horizon modifies the matrix **G** (2.11) to be of dimension $(N_2-N_1+1)\times NU$

  $$\mathbf{G}=\begin{bmatrix} g_{N_1-1} & \cdots & g_0 & 0 & 0 & \cdots & 0 \\ g_{N_1} & \cdots & g_1 & g_0 & 0 & \cdots & 0 \\ \vdots & \ddots & & & & & \vdots \\ g_{N_2-1} & \cdots & & & & & g_{N_2-NU} \end{bmatrix} \tag{2.13}$$

  which means that the matrix to be inverted $[\mathbf{G}^{\mathbf{T}}\mathbf{G}+\lambda\mathbf{I}]$ is itself of dimension $NU \times NU$. If $NU=1$, this reduces to a scalar computation.

- *Stable control of NMP processes*: The assumption of a control horizon represents a constraint upon the control calculation that allows stable control of non-minimum phase processes without the use of λ as required by some of the predictive control algorithm (for instance the GMV algorithm). The control signal is prevented from growing uncontrollably and thus unstable modes are suppressed (Clarke [11]). Of course, the restriction of using non-zero λ for processes with physical dead time $d \geq N_1$ is not necessary, for appropriate NU (Mohtadi [35]).

NU is a GPC design parameter, a large value of which implies more degrees of freedom and thus potentially more active control and a small value implies relatively conservative controller.

2.3 Basic Multivariable GPC

So far the basic SISO GPC algorithm has been discussed, in this chapter. In this section, the multivariable generalized predictive control (MGPC) algorithm will be developed, by drawing analogy with the SISO GPC, for an n-output m-input system. Details of multivariable extension of GPC can be found in Mohtadi [35] and Shah *et al.* [46].

2.3.1 The MV CARIMA Process Model

Consider a CARIMA process model for a n-output m-input system represented by

$$\mathcal{A}(q^{-1})\Delta\mathbf{y}(t) = \mathcal{B}\Delta\mathbf{u}(t-1) + \mathcal{C}\mathbf{e}(t) \tag{2.14}$$

where $\mathcal{A}$ and $\mathcal{C}$ are diagonal polynomial matrices of dimension $n \times n$

$\mathcal{B}$ is a polynomial matrix of dimension $n \times m$

$\Delta\mathbf{y}$ and $\mathbf{e}$ are the differenced output and noise vectors respectively, of dimension $n \times 1$

$\Delta\mathbf{u}$ is a differenced input vector of dimension $m \times 1$

If any of the 'input-output channels' of a process have a non-zero dead-time, the leading elements of the corresponding polynomial in matrix $\mathcal{B}$ are zero. For the MGPC, the outputs, inputs and noise vectors are written in bold to differentiate them from scalar. Any multivariable process can be represented by the above CARIMA model (equation 2.14). The MV CARIMA process model has all the properties of SISO CARIMA model, which are elaborated in section 2.2.1. In this section, for the sake of simplicity, the coefficient matrix of the noise vector, $\mathcal{C}$, has been set to the Identity matrix and its implications will be discussed in section 2.4.1.

2.3.2 Long Range Prediction in MGPC

The development of the prediction equation that predicts the outputs of a multivariable process is achieved via the matrix Diophantine equation shown below:

$$I = \mathcal{E}\mathcal{A}\Delta + q^{-j}\mathcal{F} \tag{2.15}$$

When combined with the process model of 2.14, this gives:

$$\hat{\mathbf{y}}(t+j) = \mathcal{G}_j(q^{-1})\Delta\mathbf{u}(t+j-1) + \mathcal{F}_j(q^{-1})\mathbf{y}(t) \tag{2.16}$$

where $\mathcal{G}_j = \mathcal{E}_j\mathcal{B}$. This can be rewritten as:

$$\begin{aligned} \hat{\mathbf{y}}(t+j) &= \tilde{\mathcal{G}}_j(q^{-1})\Delta\mathbf{u}(t+j-1) + \bar{\mathcal{G}}_j(q^{-1})\Delta\mathbf{u}(t-1) + \mathcal{F}_j(q^{-1})\mathbf{y}(t) && (2.17) \\ &= \tilde{\mathcal{G}}_j(q^{-1})\Delta\mathbf{u}(t+j-1) + \mathbf{f}(t+j) && (2.18) \end{aligned}$$

where, just as in the SISO case, the prediction $\hat{\mathbf{y}}(t+j)$ can be split into two distinct parts:

- **f**, the prediction of the output assuming no future changes in the control signal. This is also referred to as the 'free-response'.
- $\tilde{\mathcal{G}}_j(q^{-1})\Delta\mathbf{u}(t+j-1)$, the prediction of the output that is dependent upon the future control increments yet to be calculated.

2.3.3 The Multivariable Predictive Control Law

The cost function minimized by the multivariable GPC is given by:

$$\begin{aligned} \min J_{MGPC} &= \sum_{j=N_1}^{N_2} [\hat{\mathbf{y}}(t+j) - \mathbf{w}(t+j)]^T [\hat{\mathbf{y}}(t+j) - \mathbf{w}(t+j)] \\ &\quad + \sum_{j=1}^{NU} \Delta\mathbf{u}(t+j-1)^T \Lambda \Delta\mathbf{u}(t+j-1) \end{aligned} \tag{2.19}$$

where $\mathbf{w}$ is a vector of set-points, and Λ is a diagonal matrix enabling specification of different values of control weighting for each input independently:

$$\Lambda = \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_m \end{bmatrix} \tag{2.20}$$

The cost function of MGPC is similar to the cost function of the SISO GPC (2.8). As in SISO GPC, it is possible to calculate the $\hat{\mathbf{y}}(t+j)$ simply by iterating the estimated plant model for 'free response' of each output, using the MV CARIMA process model (2.14). The elements of the polynomial matrix, $\tilde{\mathcal{G}}$, in equation (2.18), correspond to the appropriate elements of the step-responses of individual channels.

Following the standard derivation of MGPC in Mohtadi, [35] and Shah *et al.* [46] leads to the multivariable control law given by the following cost function:

$$\min J_{MGPC} = [\mathcal{G}\tilde{U} + \mathbf{f} - \mathbf{w}]^T[\mathcal{G}\tilde{U} + \mathbf{f} - \mathbf{w}] + \Delta\mathbf{u}^T \underline{\Lambda} \Delta\mathbf{u} \tag{2.21}$$

and its solution is:

$$\Delta\mathbf{u}(t) = [I_m\ 0\ \ldots][\mathcal{G}^T\mathcal{G} + \underline{\Lambda}]^{-1}\mathcal{G}(\mathbf{w} - \mathbf{f}) \tag{2.22}$$

where

$$\begin{aligned} \tilde{U} &= [\Delta\mathbf{u}(t)\ \Delta\mathbf{u}(t+1)\ \ldots\ \Delta\mathbf{u}(t+NU-1)] \\ \mathbf{w} &= [w_1(t+1)\ \ldots\ w_n(t+1)\ \ldots\ w_1(t+j)\ \ldots\ w_n(t+j)]^T \\ \mathbf{f} &= [f_1(t+1)\ \ldots\ f_n(t+1)\ \ldots\ f_1(t+j)\ \ldots\ f_n(t+j)]^T \\ \underline{\Lambda} &= \text{matrix of } NU \text{ block diagonal } \Lambda \text{ matrices} \end{aligned}$$

If the control horizon NU is assumed equal for all channels, then the $\mathcal{G}$ matrix of dimension $n(N_2 - N_1 + 1) \times (mNU)$:

$$\mathcal{G} = \begin{bmatrix} G_{N_1-1} & \cdots & G_0 & 0 & 0 & \cdots & 0 \\ G_{N_1} & \cdots & G_1 & G_0 & 0 & \ddots & 0 \\ \vdots & & & & & & \vdots \\ G_{N_2-1} & \cdots & & & & & G_{N_2-NU} \end{bmatrix} \tag{2.23}$$

where each G_i is a submatrix of dimension $n \times m$, the elements of which are the step-responses of i_{th} sampling interval of each individual input-output relations. Since GPC is a receding-horizon control strategy only the first control increment is implemented for each input, hence the pre-multiplication by $[I_m \ 0 \ldots]$ in (2.22).

The control calculation for the MGPC proceeds in exactly the same fashion as SISO algorithm, the only difference being the increased order of calculation at each stage with polynomial matrices replacing polynomials, vectors replacing scalars etc. The properties of this MPC algorithm for the various 'tuning knobs' are in general the same as the corresponding features of the SISO algorithm. If the GPC design parameters corresponding to each loop are specified the same, then the symmetry of the control law is retained and input/output ordering or pairing has no effect upon the control performance.

2.4 Extensions to the Basic GPC Algorithm

In the preceding sections, the basic fundamentals concepts of GPC algorithm were described and the basic equations for the calculation of control signal were developed. Though this algorithm forms the main part of the control law, it is not capable of effectively controlling all real life situations. By incorporating some more design parameters into the algorithm, it can be made more robust and capable of handling real life physical systems. In the next few subsections, incorporation of the following design parameters in the controller will be considered: a polynomial for tailoring the controlled response to load disturbances, feedforward compensation, output-weighting factors, and constraints on the inputs and outputs of the process.

2.4.1 T(q^{-1}): Load Disturbance Rejection Polynomial

The term $C(q^{-1})\xi(t)/\Delta$, in the CARIMA plant model description (2.1), is assumed to describe the noise affecting the plant as a zero-mean *white* noise sequence $\{\xi(t),\ t = 1, 2, \ldots\}$ that is *colored* by $C(q^{-1})$ and then *integrated*. This will never represent an exact description of the noise structure. But this kind of representation is of practical importance as it results in a control law with inherent integral action.

The relation between the $C(q^{-1})$ and $T(q^{-1})$ is well illustrated by the following filtered CARIMA model:

$$\frac{A(q^{-1})}{T(q^{-1})}\Delta y(t) = \frac{B(q^{-1})}{T(q^{-1})}\Delta u(t-1) + \frac{C(q^{-1})}{T(q^{-1})}\xi(t) \tag{2.24}$$

Now if $T(q^{-1}) = C(q^{-1})$ then the prediction errors resulting will have the minimum variance possible for this particular model structure. But because of the time-varying nature of C, on-line estimation of C is difficult. Also, the exact nature of $\xi(t)$ is never known and must be approximated. For these reasons, a fixed estimate of C, the T polynomial, is used for control calculations.

The incorporation of the T polynomial in the algorithm results in the use of the modified Diophantine equation given by:

$$T = E_j A\Delta + q^{-j}F_j \tag{2.25}$$

and the modified prediction equation

$$\hat{y}'(t+j|t) = G_j\Delta u'(t+j-1) + F_j y'(t) \tag{2.26}$$

where the nomenclature $\Delta u'$ and y' denote that these quantities have been filtered by $1/T$.

The discussion above was for incorporation of $T(q^{-1})$ in the control algorithm for SISO GPC. The same equations hold true for MGPC with the scalars, polynomials etc. replaced by appropriate vectors and polynomial matrices. The revised Diophantine equation for the MGPC is

$$\mathcal{T} = \mathcal{E}_j\mathcal{A}\Delta + q^{-j}\mathcal{F}_j \tag{2.27}$$

where $\mathcal{T}$ is a diagonal observer polynomial matrix

$$\mathcal{T}(q^{-1}) = \text{diag}[T_1 \ \ldots \ T_n] \tag{2.28}$$

The prediction equation for MGPC is modified as:

$$\hat{\mathbf{y}}(t+j) = \mathcal{T}^{-1}[\mathcal{G}_j(q^{-1})\Delta\mathbf{u}(t+j-1) + \mathcal{F}_j(q^{-1})\mathbf{y}(t)] \tag{2.29}$$

From a practical view point, this design parameter can be described as a filter which helps in handling high frequency disturbances in the system, which can be due to the unmodelled dynamics or the presence of high frequency unmeasurable load disturbances. The use of this low-pass filter polynomial in the controller design enhances the robustness of the control and gives better load disturbance rejection properties. The property of this tuning parameter to reject load disturbances independent of servo performance, in the ideal case when there is no model process mismatch, led to the $T(q^{-1})$ being called the *load disturbance tailoring polynomial.* This polynomial is also called as T-filter in the control literature.

2.4.2 Feedforward Compensation

In the process control industry, there are many physical systems with measurable load disturbances. Instead of relying on the load disturbance properties of the controller, use of measurable load disturbances in the control strategy can substantially enhance the performance of the closed-loop system, particularly for systems with significant time-delay and/or sluggish response.

For SISO GPC feedforward can be incorporated by extending the CARIMA model *i.e.*

$$A(q^{-1})\Delta y(t) = B(q^{-1})\Delta u(t-1) + D(q^{-1})\Delta \nu(t-1) + \xi(t) \tag{2.30}$$

where $\nu(t)$ is the measurable disturbance; $D(q^{-1})$ is a polynomial in backward shift operator that may be estimated in the same manner as A or B and thus enable dynamic feedforward compensation.

Alternatively, feedforward in SISO system can be viewed as an extended multi-input single-output (MISO) system. For a MIMO systems with n-outputs and m-inputs, d-disturbance variables will result in using an extended CARIMA model of n-outputs and $(m+d)$-inputs. This extended model should be used for calculating the 'free-response' (sections 2.2.2 and 2.3.2) of the system. However, the actual CARIMA process model should be used for the remaining parts of the calculation *i.e.* for forming the step-response matrix (G or $\mathcal{G}$), and assuming m manipulated variables, etc. For SISO systems, this result in the same law as given by equation 2.30.

2.4.3 Constraints

The control action implemented on a physical system has limitations due to saturation of actuators; for example any valve can only open between the range of 0% and 100% or for example the reflux ratio of a distillation column cannot be negative. It is possible to incorporate the physical constraints of the process in the control algorithm while calculating the control signals by minimizing the cost function (2.19) subject to the physical constraints.

The usual constraints of real life processes are constraints on amplitude and/or rate of change of manipulated variables and constraints on process output. These physical constraints of a process should be mapped as constraints on the manipulated variables of the system, which for the cost function defined by (2.19) is $\Delta\mathbf{u}$.

The amplitude constraints on the process inputs can be rewritten as constraints on the $\Delta\mathbf{u}$ as follows for $NU = 1$:

$$\mathbf{u}_{min} \leq \mathbf{u}(t) \leq \mathbf{u}_{max} \tag{2.31}$$

$$\Rightarrow \mathbf{u}_{min} - \mathbf{u}(t-1) \leq \Delta\mathbf{u}(t) \leq \mathbf{u}_{max} - \mathbf{u}(t-1) \tag{2.32}$$

The amplitude constraint for a process with NU as control horizon, applied to a MIMO system will be:

$$\begin{bmatrix} \mathbf{u}_{min} - \mathbf{u}(t-1) \\ \mathbf{u}_{min} - \mathbf{u}(t-1) \\ \vdots \\ \mathbf{u}_{min} - \mathbf{u}(t-1) \end{bmatrix} \leq \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 1 & 1 & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} \Delta\mathbf{u}(t) \\ \Delta\mathbf{u}(t+1) \\ \vdots \\ \Delta\mathbf{u}(t+NU-1) \end{bmatrix} \leq \begin{bmatrix} \mathbf{u}_{max} - \mathbf{u}(t-1) \\ \mathbf{u}_{max} - \mathbf{u}(t-1) \\ \vdots \\ \mathbf{u}_{max} - \mathbf{u}(t-1) \end{bmatrix} \tag{2.33}$$

where in the above equation all the vectors are of the dimension $(m \times NU) \times 1$ and the lower diagonal matrix is of dimension $(m \times NU) \times (m \times NU)$.

The rate constraints are constraints on $\Delta\mathbf{u}$ and hence can be used directly. The constraints on the process output also should be mapped as constraints on $\Delta\mathbf{u}$. This can be done by using equation 2.19

$$\mathbf{y}_{min} \leq \hat{\mathbf{y}} \leq \mathbf{y}_{max} \tag{2.34}$$

$$\Rightarrow \mathbf{y}_{min} - \mathbf{f} \leq \mathcal{G}\Delta\mathbf{u} \leq \mathbf{y}_{max} - \mathbf{f} \tag{2.35}$$

where all the above vectors and matrices are vectorized for appropriate minimum, maximum and control horizons. However, it must be kept in mind that the mapping of constraints on process output depend on the exactness of the model. If the model is poor, the mapping of output constraints as constraints on $\Delta\mathbf{u}$ will not achieve the required effect.

The solution to the constrained GPC is achieved by minimizing the cost function (equation 2.9 or 2.21) subject to the inequality constraints described above. Commercial optimization software packages like QPSOL [19] and TOLMIN [41] are available in the market which solve such problems.

Implementing constrained control algorithm is computationally more intensive than the unconstrained case. For the general case, analytical solution of the constrained cost function does not exist and is calculated by the use of commercial software packages, which use efficient search algorithms of different kinds. For some special cases, it is possible to solve the constrained GPC algorithm analytically. Tsang and Clarke [55] have developed analytic solutions for rate or amplitude (but not both) constrained SISO GPC for the special cases of $NU = 1$ and 2. In chapter 3, analytical solutions are developed for rate and/or amplitude constrained SISO GPC for $NU = 1$ and 2 and for rates and/or amplitudes constrained MGPC for $NU = 1$. The implementation of an analytical solution is computationally much more efficient than an algorithmic solution.

To sum up, the constraints of a physical process can be incorporated in the control algorithm, which results in the calculation of physically realizable control signals. The cost paid for incorporating the constraints in the controller is additional computational load.

2.4.4 Output Weighting

This tuning knob is specific only for multivariable systems. This parameter is very important, as will be clear from later chapters (4, 5, 6). The quadratic cost-function of multivariable GPC only considers set-point tracking and *does not specifically consider decoupling.* This can be illustrated by rearranging the MGPC cost function (2.19) for a 2 × 2 example:

$$\min J_{MGPC} = \sum_{j=N_1}^{N_2} \overbrace{[\hat{y}_1(t+j) - w_1(t+j)]^2}^{loop1} + \sum_{j=N_1}^{N_2} \overbrace{[\hat{y}_2(t+j) - w_2(t+j)]^2}^{loop2}$$
$$+ \sum_{j=1}^{NU} \Delta \mathbf{u}(t+j-1)^T \Lambda \Delta \mathbf{u}(t+j-1) \qquad (2.36)$$

From the above equation, it can be seen that the control calculation depends upon the relative magnitude of $\hat{y}_1$ and $\hat{y}_2$. Exact decoupling can be achieved only if 2.36 can be brought to zero, something that will rarely be the case in practice. The practical significance of this observation is two-fold:

- To ensure 'symmetric' control performance, individual loops of MGPC should be scaled such that under typical operating conditions, the values used within the control calculation after analog to digital conversion scaling and/or normalization are of similar magnitude.
- Decoupling of a particular loop may be enhanced by weighting the corresponding output significantly more than the others, thus decoupling that output from set-point changes in the other loops (Maurath *et al.*[34]).

As will be shown in later chapters (4, 5), this 'tuning knob' also affects the performance of a multivariable system in the presence of model process mismatch.

2.5 Long Range Predictive Identification Algorithm

So far in this chapter, the generalized predictive control algorithm has been developed for SISO and MIMO systems. The adaptive GPC algorithm was implemented for some of the experimental runs, on a SISO process, described in chapter 5. Long Range predictive identification (LRPI) scheme of Shook *et al.* [47,48,49] was used for identifying the parameters of the process, for these experimental runs.

The LRPI algorithm uses a multi-step cost function for parameter identification and is a *dual* of the long range predictive control schemes. The closed-loop performance of this multi-step identification-prediction algorithm is better than algorithms using a single-step ahead estimation and these parameters to obtain multi-step ahead prediction. The cost function of the LRPI is:

$$J_{LRPI} = \sum_{t=1}^{N} \sum_{j=N_1}^{N_2} (y(t) - \hat{y}(t|t-j))^2 \tag{2.37}$$

The configuration of the LRPI and GPC in the closed loop is shown in figure 2.1. The reader is referred to [48,49] for the mathematical details and development of the LRPI algorithm.

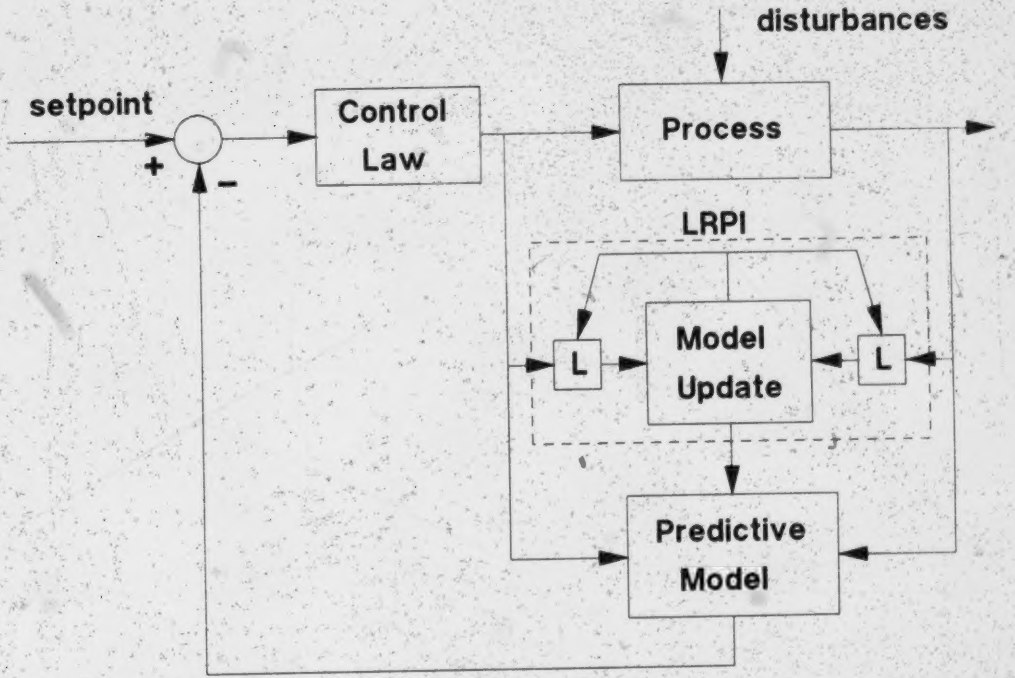

disturbances
setpoint
+
−
Control Law
Process
LRPI
L
Model Update
L
Predictive Model

Figure 2.1: Configuration of LRPI and GPC in the Closed Loop

Chapter 3

Analytical Solution for Simple Cases of Constrained GPC

An analytical solution exists for the unconstrained Generalized Predictive Control algorithm (equation 2.22). But no analytical solution exists for the general case of constrained GPC. The solution for the general case of constrained GPC can be found by using commercial optimization software packages. The on-line use of optimization software packages is computationally intensive and should be avoided as far as possible. However, the unconstrained GPC algorithm cannot account for the physical constraints of the actuators in a process. To accommodate the physical constraints and calculate signals which are physically realizable by the process, it is necessary to implement constrained GPC algorithm. The formulation of constrained GPC has been discussed in section 2.4.3.

Albeit no analytical solution exists for the general case of the cost function of the constrained GPC algorithm, analytical solutions exist for some simple cases of GPC and are of practical importance. The computation required by these analytic solutions is substantially lower than that required by algorithmic solutions and hence are more efficient. In this chapter, an analytical solution for the rate and amplitude constrained GPC will be considered for the special cases of $NU = 1$ and 2 for a SISO system and $NU = 1$ for a MIMO system.

3.1 Development of Analytical Solution

3.1.1 Introduction

The GPC algorithm can be subjected to rate and amplitude constraints on the manipulated variable, and can be used for systems with physical limitations on the implemented control. The solution of the objective function of the constrained GPC can be calculated on-line with the help of efficient quadratic programming(QP) optimization software packages which are computationally very intensive. Therefore, the use of an analytical solution to solve the cost function of constrained GPC, wherever possible, is highly desirable as it requires reduced computation. Unlike most of the algorithmic solutions, which may or may not converge to the true solution, analytical solutions do not suffer from this problem.

Tsang and Clarke [55] have developed an algorithm for SISO systems, with $NU \leq 2$, to calculate the control signal for the rate or amplitude constrained GPC, but not both rate and amplitude constraints simultaneously. This chapter develops an algorithm for computing the analytical solution of GPC with constraints on rate and amplitude of the manipulated variable(s), for the special cases of NU equal to 1 and 2 for a SISO system and NU equal to 1 for a MIMO system. A geometric illustration of constraints and their effect on the solution of the objective function is also provided.

The optimal solution of the unconstrained cost function solution will be shown with the superscript '$*$' and the optimal solution to the constrained cost function will be shown with the superscript '$+$' in the rest of the chapter. The analytical solution of the constrained GPC was realized using Kuhn-Tucker multipliers method, which is described in the following section.

3.1.1.1 Kuhn Tucker Multipliers Method

This method can be used for solving non-linear cost functions with linear constraints. The details of the algorithm are described in Singh and Titli [50]. Briefly, the algorithm can be explained as follows:

Consider the minimization of a function subject to constraints, *i.e.*

$$\min f(x) \tag{3.1}$$

such that

$$\begin{aligned} a_i(x) &\leq 0 \quad \text{for } i = 1 \text{ to } p \\ b_j(x) &= 0 \quad \text{for } j = 1 \text{ to } q \end{aligned}$$

The cost function of GPC is a quadratic function. Hence the cost function of $f(x)$ for the quadratic programming (QP) problem can be stated in the following form:

$$J_{QP} = \min \quad 2c^T x + x^T H x \tag{3.2}$$

The constrained problem of equation 3.1 can rewritten by including the constraints as an unconstrained problem as follows:

$$\Phi(\underline{x}, \lambda, \mu) = f(\underline{x}) + \sum_{i=1}^{p} \mu_i a_i(\underline{x}) + \sum_{j=1}^{q} \lambda_j b_j(\underline{x}) \tag{3.3}$$

where the μ_i's are the Kuhn-Tucker multipliers and the λ_j's are the Lagrangian multipliers. For Φ to be minimum, the following first order and second order conditions must be satisfied. The first order conditions are:

- $\underline{\Phi}_x = \underline{0}$
- $\underline{\Phi}_\lambda = \underline{b}(\underline{x}) = \underline{0}$
- $\underline{\mu}^T \underline{a}(\underline{x}) = 0$

where the notation is $\Phi_\lambda = \frac{\partial \Phi}{\partial \lambda}$.

The second order conditions for Φ in equation 3.3 to be minimum are:

- $\mu_i \geq 0$ for $i = 1$ to p
- Φ_{xx} is non-negative definite in the subspace of the constraints.

In the above algorithm, it must be noted that the condition on the sign of the Kuhn-Tucker multipliers μ_i changes with the nature of problem as shown in table 3.1.

Nature of the constraint	Nature of the extremum	
	Max $f(x)$	Min $f(x)$
subject to $a_j(x) \leq 0$	$\mu_i \leq 0$	$\mu_i \geq 0$
subject to $a_j(x) \geq 0$	$\mu_i \geq 0$	$\mu_i \leq 0$

Table 3.1: Condition on Kuhn-Tucker multipliers for different formulations of the problem

This algorithm is very similar to method of Lagrange multipliers and some authors also refer to the Kuhn-Tucker multiplier as the Lagrange multiplier. For a more detailed explanation of the algorithm, the reader is referred to Singh and Titli [50].

3.1.2 Constrained SISO GPC for $NU = 1$

This is the simplest case of constrained GPC. From equation 2.8, the cost function of the unconstrained GPC for $NU = 1$ is:

$$J_{GPC} = \sum_{j=N_1}^{N_2} [y(t+j) - w(t+j)]^2 + \lambda \Delta u(t)^2 \tag{3.4}$$

The constraints on the rate and amplitude of the manipulated variable are:

$$u_{min} \leq u(t) \leq u_{max} \tag{3.5}$$

$$\Delta u_{min} \leq \Delta u(t) \leq \Delta u_{max} \tag{3.6}$$

As the cost function (equation 3.4) is in terms of the variable Δu, all the constraints must be translated as constraints on Δu. The amplitude constraint can be mapped as constraint on Δu, as explained in section 2.4.3.

The solution of the cost function (equation 3.4) subject to the constraints given by equations 3.5 and 3.6 can be calculated as follows. Define:

$$\alpha = \max\{\Delta u_{min}, u_{min} - u(t-1)\} \tag{3.7}$$

$$\beta = \min\{\Delta u_{max},\ u_{max} - u(t-1)\}$$

The solution $(\Delta u^*(t))$ for the unconstrained GPC can be calculated using the equation 2.10. Then the solution of the constrained GPC is given by clipping the unconstrained $\Delta u^*(t)$ to the appropriate bound *i.e.*

$$\Delta u^+(t) = \begin{cases} \alpha & \text{if} \quad \Delta u^*(t) \leq \alpha \\ \Delta u^*(t) & \text{if} \quad \alpha \leq \Delta u^*(t) \leq \beta \\ \beta & \text{if} \quad \beta \leq \Delta u^*(t) \end{cases} \tag{3.8}$$

3.1.3 Constrained SISO GPC for $NU = 2$

In this section, the solution to the rate and amplitude constrained SISO GPC will be developed for $NU = 2$. The Kuhn-Tucker's multiplier method described in section 3.1.1.1 is used for finding the solution of the constrained problem. The vectorized cost function of the unconstrained GPC (equation 2.9) is:

$$\begin{aligned} \min J_{GPC} &= [\mathbf{G}\tilde{\mathbf{u}} + \mathbf{f} - \mathbf{w}]^T[\mathbf{G}\tilde{\mathbf{u}} + \mathbf{f} - \mathbf{w}] + \lambda\tilde{\mathbf{u}}^T\tilde{\mathbf{u}} && (3.9) \\ &= \tilde{\mathbf{u}}^T(\mathbf{G}^T\mathbf{G} + \lambda I)\tilde{\mathbf{u}} + 2(\mathbf{f} - \mathbf{w})^T\mathbf{G}\tilde{\mathbf{u}} + (\mathbf{f} - \mathbf{w})^T(\mathbf{f} - \mathbf{w}) && (3.10) \end{aligned}$$

where, for $NU = 2$:

$$\begin{aligned} \mathbf{w} &= [w(t+N_1)\ w(t+N_1+1)\ \ldots\ w(t+N_2)]^T \\ \mathbf{f} &= [f(t+N_1)\ f(t+N_1+1)\ \ldots\ f(t+N_2)]^T \\ \tilde{\mathbf{u}} &= [\Delta u(t)\ \ \Delta u(t+1)]^T \end{aligned}$$

$$\mathbf{G} = \begin{bmatrix} g_{N_1-1} & g_{N_1-2} \\ g_{N_1} & g_{N_1-1} \\ \vdots & \vdots \\ g_{N_2-1} & g_{N_2-2} \end{bmatrix}$$

For the constrained problem, the cost function of equation 3.10 must be subjected to rate and amplitude constraints. The constraints can be written as:

$$\begin{aligned} u_{1min} &\leq u(t) \leq u_{1max} \\ u_{2min} &\leq u(t+1) \leq u_{2max} \\ \Delta u_{1min} &\leq \Delta u(t) \leq \Delta u_{1max} \\ \Delta u_{2min} &\leq \Delta u(t+1) \leq \Delta u_{2max} \end{aligned} \tag{3.11}$$

The subscripts 1 and 2 for the rate and amplitude constraints are for the sake of generality. Depending on the application, the values of minimum and maximum constraints u_1 and Δu_1 can be equal to u_2 and Δu_2 respectively. To apply the Kuhn-Tucker multipliers method for solving equation 3.10 subject to the constraints in equation 3.11, the constraints should be rewritten as:

$$\begin{aligned} a_1 &= \Delta u(t) - \alpha_1 \geq 0 \\ a_2 &= -\Delta u(t) + \beta_1 \geq 0 \\ a_3 &= \Delta u(t) + \Delta u(t+1) - \alpha_2 \geq 0 \\ a_4 &= -\Delta u(t) - \Delta u(t+1) + \beta_2 \geq 0 \\ a_5 &= \Delta u(t+1) - \alpha_3 \geq 0 \\ a_6 &= -\Delta u(t+1) + \beta_3 \geq 0 \end{aligned} \tag{3.12}$$

where

$$\begin{aligned} \alpha_1 &= \max(\Delta u_{1min}, u_{1min} - u(t-1)) \\ \beta_1 &= \min(\Delta u_{1max}, u_{1max} - u(t-1)) \\ \alpha_2 &= u_{2min} - u(t-1) \\ \beta_2 &= u_{2max} - u(t-1) \\ \alpha_3 &= \Delta u_{2min} \\ \beta_3 &= \Delta u_{2max} \end{aligned} \tag{3.13}$$

Solution of Constrained SISO GPC for $NU = 2$ In this section, the solution of the cost function (equation 3.10) subject to the constraints (equation 3.12) will be developed.

Define $\mathbf{A}$ as the set of constraints in equation 3.12:

$$\mathbf{A} = [a_1\ a_2\ a_3\ a_4\ a_5\ a_6]^T \tag{3.14}$$

The corresponding Kuhn-Tucker multiplier for the constraints are:

$$\mu = [\mu_1\ \mu_2\ \mu_3\ \mu_4\ \mu_5\ \mu_6]^T \tag{3.15}$$

The unconstrained cost function 3.10 can be augmented with the constraints,$\mathbf{A}$ using the Kuhn-Tucker multipliers, μ as:

$$\min J = \tilde{\mathbf{u}}^T(\mathbf{G}^T\mathbf{G} + \lambda I)\tilde{\mathbf{u}} + 2(\mathbf{f} - \mathbf{w})^T\mathbf{G}\tilde{\mathbf{u}} + (\mathbf{f} - \mathbf{w})^T(\mathbf{f} - \mathbf{w}) + \mu^T\mathbf{A} \tag{3.16}$$

Let:

$$\begin{aligned} \mathbf{G}^T\mathbf{G} + \lambda I &= H_{NU\times NU} = \begin{bmatrix} h_{11} & h_{12} \\ h_{12} & h_{22} \end{bmatrix} \\ (\mathbf{f} - \mathbf{w})^T\mathbf{G} &= c^T = [c_1\ \ c_2] \end{aligned} \tag{3.17}$$

Note that the term $(\mathbf{f} - \mathbf{w})^T(\mathbf{f} - \mathbf{w})$ in equation 3.16 can be dropped as it is independent of $\tilde{\mathbf{u}}$. The matrix H in 3.17 is the Hessian (matrix of second partial derivative) of the cost function. With the above definitions, equation 3.16 can be rewritten as:

$$\begin{aligned} \min J &= \tilde{\mathbf{u}}^T H \tilde{\mathbf{u}} + 2\, c^T \tilde{\mathbf{u}} + \mu\mathbf{A} \\ &= h_{11}\Delta u(t)^2 + 2h_{12}\Delta u(t)\Delta u(t+1) + h_{22}\Delta u(t+1)^2 \\ &\quad + 2c_1\Delta u(t) + 2c_2\Delta u(t+1) + \mu^T\mathbf{A} \end{aligned} \tag{3.18}$$

The Kuhn-Tucker multipliers method requires that for $\tilde{\mathbf{u}}$ to be optimal, it should satisfy the following conditions:

1. The first order conditions are:

$$\partial J/\partial\tilde{\mathbf{u}} = 0$$

2. The second order conditions are:

$$\begin{aligned} a_i\mu_i = 0 \text{ for all } i = 1 \text{ to } 6 \text{ such that } \quad & a_i = 0 \quad \text{if} \quad \mu_i < 0 \\ \text{or} \quad & a_i > 0 \quad \text{if} \quad \mu_i = 0 \end{aligned}$$

From the first order condition $\partial J/\partial \tilde{\mathbf{u}}$, we have:

$$\frac{\partial J}{\partial \Delta u(t)} = 2h_{11}\Delta u(t) + 2h_{12}\Delta u(t+1) + 2c_1 + \mu_1 - \mu_2 + \mu_3 - \mu_4 = 0 \qquad (3.19)$$

$$\frac{\partial J}{\partial \Delta u(t+1)} = 2h_{12}\Delta u(t) + 2h_{22}\Delta u(t+1) + 2c_2 + \mu_3 - \mu_4 + \mu_5 - \mu_6 = 0 \qquad (3.20)$$

From these two equations (3.19, 3.20) and the Kuhn-Tucker condition, an optimal set of control moves can be calculated when the unconstrained solution violates any of the six constraints defined by equation 3.12. The solution algorithm is as follows:

Algorithm for solution of SISO GPC for $NU = 2$

Step 1: Determine the unconstrained solution.

Step 2: Test to determine if any of the constraints are violated. If all $a_i \geq 0$ (no constraints are violated) then go to **Step 5**.

Step 3: Saturate the solution on a constraint and verify if the solution satisfies the conditions of Kuhn-Tucker multipliers method *i.e.*

- assume any one or two (say a_m and a_n) of the constraints are satisfied by the constrained optimal solution, which gives a value of $\tilde{\mathbf{u}}$. Therefore, all other μ's other than μ_m and μ_n are zero.
- Calculate the Kuhn-Tucker multipliers μ_m and μ_n with the help of equations 3.19 and 3.20. Verify if the Kuhn-Tucker multipliers are ≤ 0.

If the conditions of the Kuhn-Tucker multipliers method are satisfied, then go to **Step 5**.

Step 4: Repeat **Step 3** for the next (set of) constraint(s).

Step 5: Implement the control action and repeat **Step 1–5** at next sample period.

A total of 18 sets of constraints are possible in **Step 3** of the above algorithm. They are listed in appendix A.1. It is important to note that the cost function of GPC is a quadratic and geometrically represents an ellipse on the $\Delta u(t)$—$\Delta u(t+1)$ plane and if the unconstrained solution does not lie in the feasible region of the constraints then the solution to the constrained problem lies on the boundary of the feasible region or at the intersection of two or more constraints (if a solution exists). This fact is used in the above algorithm.

3.1.4 Simulations

The objective of the simulations in this section is to demonstrate the effectiveness of constrained GPC algorithm over the standard unconstrained GPC algorithm. The continuous-time model of a distillation column (Morris *et al.*[37]) was selected for implementing the discrete-time control algorithm. The algorithm is described in section 3.1.3 and its solution is listed in appendix A.1. The simulations were performed using the software MATLAB-386.

The transfer function of the distillation column [37] controlling the top composition, x_d, with the reflux ratio, R, as the manipulated variable is:

$$\frac{X_d(s)}{R(s)} = \frac{1.0081e^{-0.35s}}{4.36s+1} \tag{3.21}$$

A sampling time of 0.35 minutes was chosen for the discrete-time control. For all the simulations, the GPC 'tuning knobs' were set to $N_1 = 1, N_2 = 6, NU = 2$, and $\lambda = 0$.

For any distillation column, the reflux ratio can be varied between 0 and ∞. It cannot be negative. Simulations were performed for unconstrained GPC, rate constrained GPC, amplitude constrained GPC and rate plus amplitude constrained GPC. In each of these cases the setpoint was a square wave of amplitudes 30 and 50 units.

Unconstrained GPC: For this simulation, the standard unconstrained GPC algorithm was used. The time-trajectory plot of this simulation is shown in figure 3.1. The set-point tracking of this simulation is ideal. But the implemented maximum control signal is about 375 and the minimum is -200. Physically negative control signals cannot be implemented and hence this control law can never be realized.

Rate Constrained GPC: The maximum and minimum rate limits on the manipulated variable were set to +50 and -50 respectively. No constraints on the amplitude of

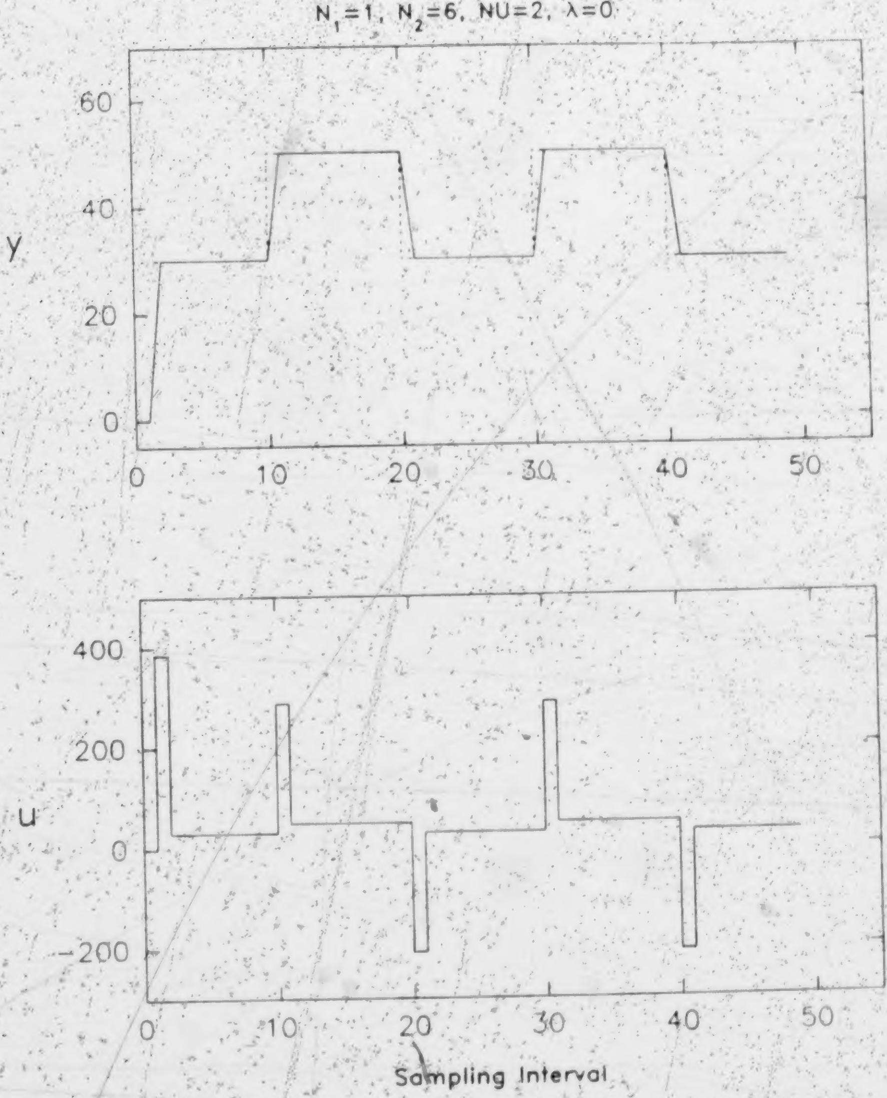


Figure 3.1: Simulation of a distillation column under unconstrained GPC

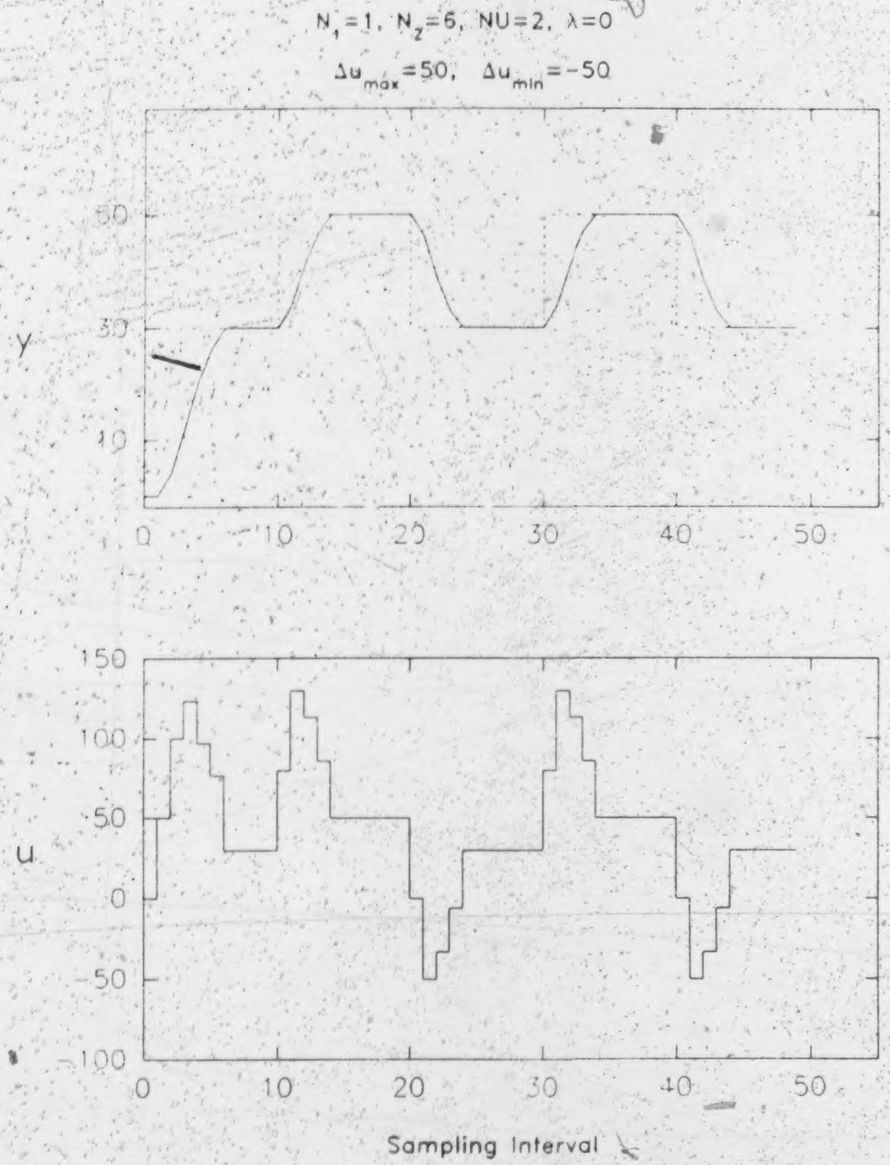


Figure 3.2: Simulation of a distillation column under rate constrained GPC

the manipulated variable were implemented. The time-trajectory plot for this case is shown in figure 3.2. Just as in the unconstrained case, the control law in this case also generates negative control signals and hence cannot be physically realized.

Amplitude Constrained GPC: The maximum and minimum amplitude constraints on the manipulated variable were set to 100 and 0 respectively. The simulation was performed without any constraints on the rate of manipulated variable. The time-trajectory plot of this case is shown in figure 3.3. The signal calculated by the control law is physically realizable, as it is always greater than or equal to zero. However, the control law is very aggressive *i.e.* some of the control moves are very large in amplitude. To overcome this problem, the control law can be detuned by incorporating rate constraints in the algorithm.

Rate and Amplitude Constrained GPC: The maximum and the minimum rate constraints were set to -50 and +50, and the maximum and the minimum amplitude constraints were set to 0 and 100 on the manipulated variable. The time-trajectory plot for this simulation is shown in figure 3.4. This control law is realizable and is less aggressive than the amplitude constrained GPC and hence is more desirable. Notice the corresponding sluggish output response as a result of this detuned control action.

The simulations demonstrate that the analytical solution for constrained GPC can be calculated and its performance is better than the unconstrained GPC algorithm. The control law given by the rate plus amplitude constrained GPC is realizable and more robust. For implementing constrained GPC, a little extra computation is required. Nevertheless, it is much lower than that required by an algorithmic solutions.

3.1.5 Constrained MIMO GPC for $NU = 1$

This section develops the solution for the rate and amplitude constrained MIMO GPC for $NU = 1$. The algorithm is similar in many respects to the algorithm developed for SISO GPC for $NU = 2$.

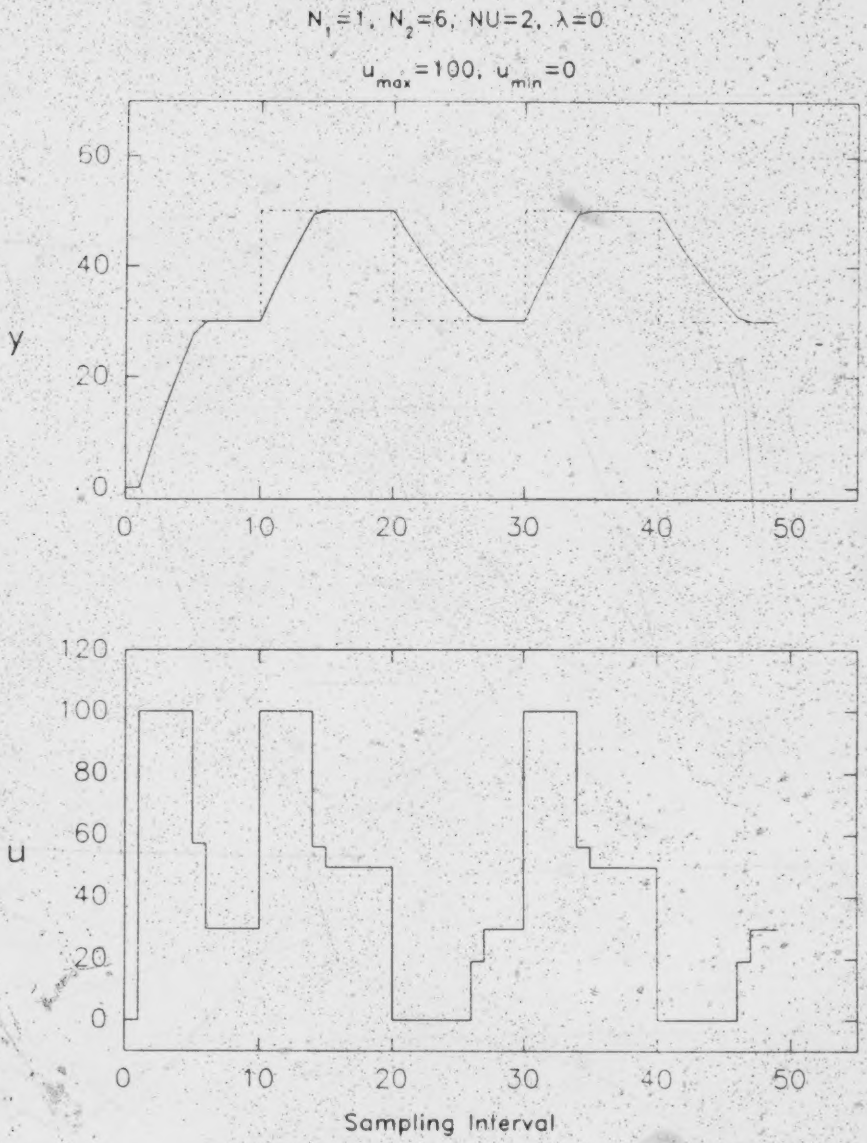


Figure 3.3: Simulation of a distillation column under amplitude constrained GPC

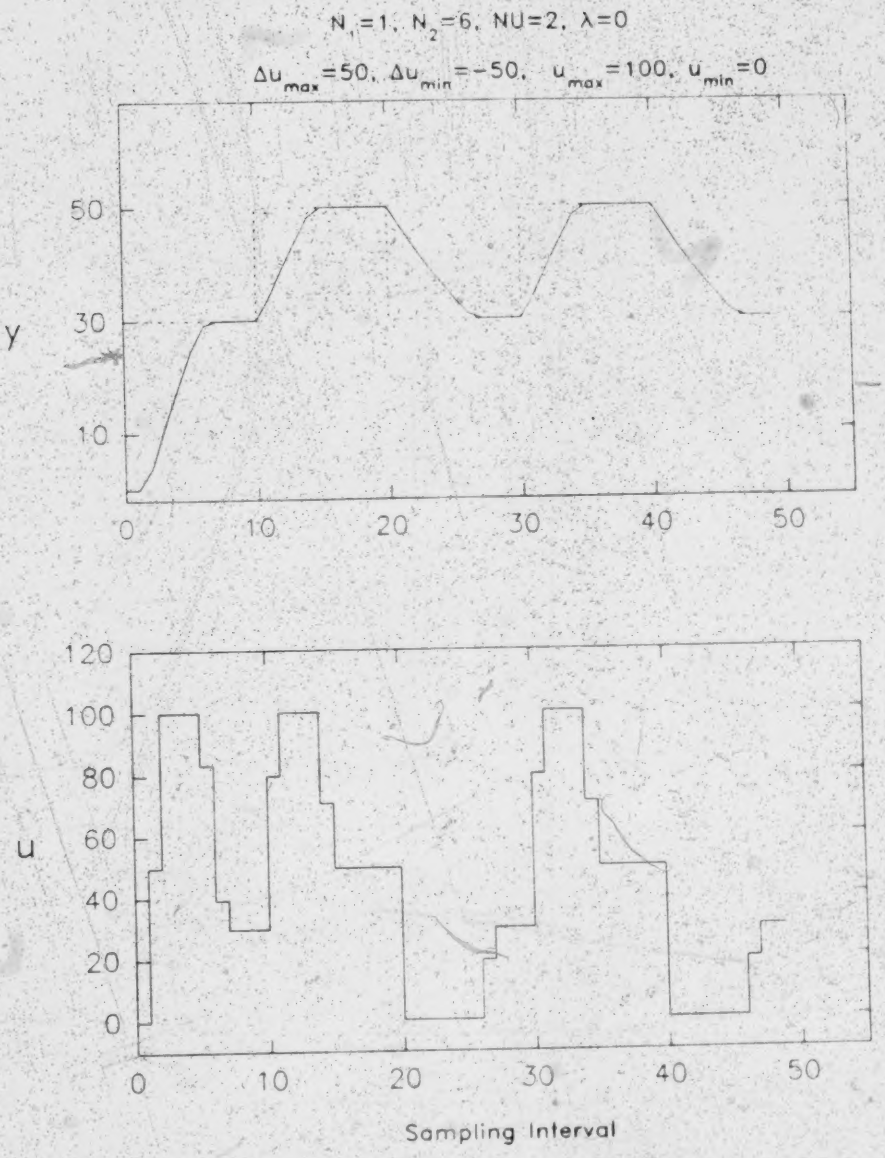


Figure 3.4: Simulation of a distillation column under rate plus amplitude constrained GPC

Recall the cost function and its solution for the unconstrained MGPC (equations 2.21 and 2.22). For $NU = 1$, the cost function is:

$$J_{MGPC} = [\mathcal{G}\Delta\mathbf{u}(t) + \mathbf{f} - \mathbf{w}]^T[\mathcal{G}\Delta\mathbf{u}(t) + \mathbf{f} - \mathbf{w}] + \Delta\mathbf{u}(t)^T\Lambda\Delta\mathbf{u}(t) \qquad (3.22)$$

and its solution is:

$$\Delta\mathbf{u}(t) = [\mathcal{G}^T\mathcal{G} + \Lambda]^{-1}\mathcal{G}^T(\mathbf{w} - \mathbf{f})$$

where:

$$\mathbf{w} = [w_1(t+1) \ldots w_n(t+1) \ldots w_1(t+j) \ldots w_n(t+j)]^T$$

$$\mathbf{f} = [f_1(t+1) \ldots f_n(t+1) \ldots f_1(t+j) \ldots f_n(t+j)]^T$$

and the $\mathcal{G}$ matrix is of dimension $(n \times (N_2 - N_1 + 1) \times m)$:

$$\mathcal{G} = \begin{bmatrix} G_{N_1-1} \\ G_{N_1} \\ \vdots \\ G_{N_2-1} \end{bmatrix}$$

where each G_i is a submatrix of dimension $n \times m$, the elements of which are the step-responses of the i_{th} sampling interval of each individual input-output relations.

For the constrained problem, the cost function of equation 3.22 must be subjected to rate and amplitude constraints. The constraints can be written as:

$$\begin{aligned} \mathbf{u}_{min} &\leq \mathbf{u}(t) \leq \mathbf{u}_{max} \\ \Delta\mathbf{u}_{min} &\leq \Delta\mathbf{u}(t) \leq \Delta\mathbf{u}_{max} \end{aligned} \qquad (3.23)$$

To apply the Kuhn-Tucker multipliers method to solve the equation 3.22 subject to the equation 3.23, the above constraints should be rewritten as:

$$\begin{aligned} \mathbf{a}_l &= \Delta\mathbf{u}(t) - \alpha \geq 0 \\ \mathbf{a}_u &= -\Delta\mathbf{u}(t) + \beta \geq 0 \end{aligned} \qquad (3.24)$$

where:

$$\begin{aligned} \alpha &= \max(\Delta\mathbf{u}_{min}, \mathbf{u}_{min} - \mathbf{u}(t-1)) \\ \beta &= \min(\Delta\mathbf{u}_{max}, \mathbf{u}_{max} - \mathbf{u}(t-1)) \end{aligned} \qquad (3.25)$$

Solution of Constrained MIMO GPC for $NU = 1$

In this section, the solution of the cost function (equation 3.22) subject to the constraints (equation 3.24) will be developed.

Note that both $\mathbf{a_l}$ and $\mathbf{a_u}$ in equation 3.24 represent m constraints, for an n-output m-input system. The corresponding Kuhn-Tucker multipliers for the constraints are:

$$\mu = [\mu_l \quad \mu_u]^T \tag{3.26}$$

where μ_l and μ_u are of dimension $m \times 1$ each.

The unconstrained cost function can be augmented with the constraints, $\mathbf{a_l}$ and $\mathbf{a_u}$ using the Kuhn-Tucker multipliers, μ as:

$$\begin{aligned} J &= \Delta\mathbf{u}(t)^T(\mathcal{G}^T\mathcal{G} + \Lambda)\Delta\mathbf{u}(t) + 2(\mathbf{f} - \mathbf{w})^T\mathcal{G}\Delta\mathbf{u}(t) \\ &\quad +(\mathbf{f} - \mathbf{w})^T(\mathbf{f} - \mathbf{w}) + \mu_l^T\mathbf{a_l} + \mu_u^T\mathbf{a_u} \\ &= \Delta\mathbf{u}(t)^T(\mathcal{G}^T\mathcal{G} + \Lambda)\Delta\mathbf{u}(t) + 2(\mathbf{f} - \mathbf{w})^T\mathcal{G}\Delta\mathbf{u}(t) \\ &\quad +\mu_l^T(\Delta\mathbf{u}(t) - \alpha) + \mu_u^T(-\Delta\mathbf{u} + \beta) \end{aligned} \tag{3.27}$$

In the above equation, the terms $(\mathbf{f} - \mathbf{w})^T(\mathbf{f} - \mathbf{w})$ has been dropped as it is independent of $\Delta\mathbf{u}(t)$ and does not affect the solution.

The Kuhn-Tucker multipliers method requires that for $\Delta\mathbf{u}(t)$ to be optimal, it should satisfy the following conditions:

- The first order conditions are:

$$\partial J/(\partial\Delta\mathbf{u}(t)) = 0$$

- The second order conditions are:

$$\begin{aligned} a_{ij}\mu_{ij} = 0 \text{ for } i = l, u \text{ and } j = 1 \text{ to } m \text{ such that} \quad & a_{ij} = 0 \quad \text{if} \quad \mu_{ij} < 0 \\ \text{or} \quad & a_{ij} > 0 \quad \text{if} \quad \mu_{ij} = 0 \end{aligned}$$

Setting the partial derivative of the cost function defined by equation 3.22 with respect to $\Delta \mathbf{u}(t)$ to zero, we have

$$\frac{\partial J}{\partial \Delta \mathbf{u}(t)} = 2(\mathcal{G}^T\mathcal{G} + \Lambda)\Delta \mathbf{u}(t) + 2\mathcal{G}^T(\mathbf{f} - \mathbf{w}) + \mu_l - \mu_u = 0 \qquad (3.28)$$

The above equation is a vectorized representation of m scalar equations. A solution of these equations satisfying the second order conditions will give the optimal solution. Therefore, the solution algorithm will be as follows:

Algorithm for solution of MIMO GPC for $NU = 1$

Step 1: Determine the unconstrained solution.

Step 2: Test to determine if any of the constraints are violated. If all $a_i \geq 0$ (no constraints are violated) then go to **Step 5**.

Step 3 Saturate a constraint and verify if the solution satisfies the conditions of Kuhn-Tucker multipliers method *i.e.*

- assume any one or more (say a_{ip} and a_{jq}, where $i, j = l, u$ and $p, q = 1$ to m) of the constraints are satisfied by the constrained optimal solution, which gives a value of $\Delta \mathbf{u}(t)$. Therefore, all other μ's other than μ_{ip} and μ_{jq} are zero.
- Calculate the Kuhn-Tucker multipliers μ_{ip} and μ_{jq} with the help of equations 3.19 and 3.20. Verify if the Kuhn-Tucker multipliers are ≤ 0.

If the Kuhn-Tucker multipliers are negative, then go to **Step 5**

Step 4 Repeat **Step 3** for the next (set of) constraint(s).

Step 5 Implement the control action and wait for the next sample period.

For **Step 3** of the above algorithm, a total of 8 sets of constraints are possible for a two-input two-output MIMO system. The solution to the two-input two-output rate and amplitude constrained MGPC is listed in appendix A.2. Similarly, for a three-input three-output system, **Step 3** of the algorithm would result in a combination of 27 sets of constraints.

3.2 Geometric Interpretation of Constrained GPC

In this section, geometric interpretations of the objective function of GPC, the constraints and the optimal solution of the controller cost function are presented for the SISO GPC with control horizon $NU = 2$. However, most of the discussion is also valid for MGPC, for a (2×2) system with $NU = 1$, when $\Delta u(t)$ and $\Delta u(t+1)$ are replaced by $\Delta u_1(t)$ and $\Delta u_2(t)$.

Subsection 3.2.1 describes the geometric shape of the cost function of GPC, subsection 3.2.2 describes the feasible region for rate, amplitude and rate plus amplitude constraints, and the final subsection 3.2.3 will interpret the optimal solution of the constrained GPC.

3.2.1 Geometric Shape of the GPC Cost Function

Consider the equation:

$$ax_1^2 + 2bx_1x_2 + cx^2 + d = 0 \tag{3.29}$$

This can also be rewritten as:

$$[\, x_1 \quad x_2 \,]\; H \;[\, x_1 \quad x_2 \,]^T + d = 0 \tag{3.30}$$

where:

$$H = \begin{bmatrix} a & b \\ b & c \end{bmatrix}$$

Equation 3.30 represents an ellipse centered at the origin if the eigenvalues of the matrix H are both of the same sign as the constant d. The cost function of GPC is defined by equation 3.10 as:

$$J_{GPC} = \tilde{\mathbf{u}}^T(\mathbf{G}^T\mathbf{G} + \lambda I)\tilde{\mathbf{u}} + 2(\mathbf{f} - \mathbf{w})^T\mathbf{G}\tilde{\mathbf{u}} + (\mathbf{f} - \mathbf{w})^T(\mathbf{f} - \mathbf{w})$$

where $\tilde{\mathbf{u}} = [\Delta u(t) \quad \Delta u)t + 1)]^T$ for $NU = 2$. By subjecting this equation to change of coordinates, it is possible to write the equation in a form similar to equation 3.30. The matrix $(G^TG + \lambda I)$ is similar to matrix H in equation 3.30. $(G^TG + \lambda I)$ is a positive definite matrix and hence all its eigenvalues are positive. For different positive values of J_{GPC}, the cost function of GPC represents loci of ellipses on $\Delta u(t)$—$\Delta u(t+1)$ plane. The objective of the unconstrained optimization problem is to find the smallest ellipse satisfying the cost function.

The eigenvalues and eigenvectors of $(G^TG + \lambda I)$ indicate the lengths of major and minor axis and their orientation on $\Delta u(t)$—$\Delta u(t+1)$ plane. Hence, the ratio of the square root of the larger to the smaller eigenvalue, which is also the condition number of the matrix, indicates whether the ellipses are oblong or circular.

3.2.2 Geometric Shapes of the Feasible Region

The region which satisfies all the constraints is defined as the feasible region. The feasible regions for rate constraints, amplitude constraints, and rate plus amplitude constraints can be shown graphically on $\Delta u(t)$—$\Delta u(t+1)$ plane, for $NU = 2$. Geometric shape of different constraints is illustrated below:

3.2.2.1 Rate Constraints

The rate constraints on the $\tilde{\mathbf{u}}$ can be represented as (equation 3.11, section 3.1.3):

$$\begin{array}{ccccc} \Delta u_{min} & \leq & \Delta u(t) & \leq & \Delta u_{1max} \\ \Delta u_{2min} & \leq & \Delta u(t+1) & \leq & \Delta u_{2max} \end{array}$$

On the $\Delta u(t)$—$\Delta u(t+1)$ plane, the constraints form a rectangular feasible region. For a given value of rate constraint, the size of the rectangular feasible region does not change with time. The feasible region for rate constraints is shown in figure 3.5.

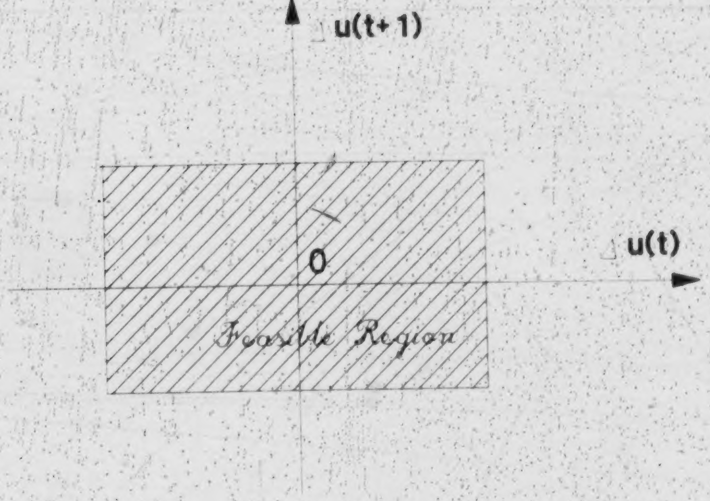


Figure 3.5: Feasible region of rate constraints

3.2.2.2 Amplitude Constraints

In section 3.1.3, it is shown that the amplitude constraints on the manipulated variable can be represented as constraints on $\tilde{\mathbf{u}}$ (equations 3.11, 3.12) as follows:

$$\begin{aligned} u_{1min} &\leq u(t) \leq u_{1max} \\ u_{2min} &\leq u(t+1) \leq u_{2max} \end{aligned}$$

which can be rewritten as:

$$\begin{aligned} \alpha_1 &\leq \Delta u(t) \leq \beta_1 \\ \alpha_2 &\leq \Delta u(t) + \Delta u(t+1) \leq \beta_2 \end{aligned} \tag{3.31}$$

where

$$\begin{aligned} \alpha_1 &= u_{1min} - u(t-1) \\ \beta_1 &= u_{1max} - u(t-1) \\ \alpha_2 &= u_{2min} - u(t-1) \\ \beta_2 &= u_{2max} - u(t-1) \end{aligned}$$

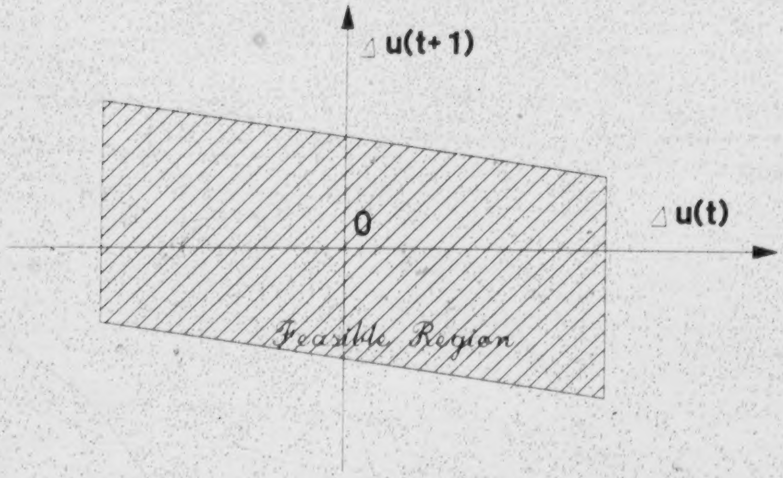


Figure 3.6: Feasible region of the amplitude constraints

The feasible region of the amplitude constraints (equation 3.31) represent a parallelogram on the $\Delta u(t)$—$\Delta u(t+1)$ plane, as shown in figure 3.6. For a given value of amplitude constraints, the size of feasible region can vary each sampling instant because each of the α_i, β_i for $i = 1,2$ is a function of $u(t-1)$. Because of this property of amplitude constraints, the solution of the amplitude constrained GPC is not trivial. However, the shape of the feasible region is always a parallelogram.

3.2.2.3 Rate plus Amplitude Constraints

In section 3.1.3, the rate plus amplitude constraints (equation 3.11) were written as:

$$\begin{array}{ccccc} u_{1min} & \leq & u(t) & \leq & u_{1max} \\ u_{2min} & \leq & u(t+1) & \leq & u_{2max} \\ \Delta u_{1min} & \leq & \Delta u(t) & \leq & \Delta u_{1max} \\ \Delta u_{2min} & \leq & \Delta u(t+1) & \leq & \Delta u_{2max} \end{array}$$

which can be rewritten as:

$$\begin{aligned} \alpha_1 &\leq \Delta u(t) \leq \beta_1 \\ \alpha_2 &\leq \Delta u(t) + \Delta u(t+1) \leq \beta_2 \\ \alpha_3 &\leq \Delta u(t+1) \leq \beta_3 \end{aligned} \tag{3.32}$$

where

$$\begin{aligned} \alpha_1 &= \max(\Delta u_{1min}, u_{1min} - u(t-1)) \\ \beta_1 &= \min(\Delta u_{1max}, u_{1max} - u(t-1)) \\ \alpha_2 &= u_{2min} - u(t-1) \\ \beta_2 &= u_{2max} - u(t-1) \\ \alpha_3 &= \Delta u_{2min} \\ \beta_3 &= \Delta u_{2max} \end{aligned}$$

The feasible region of the rate plus amplitude constraints (equation 3.32) is represented by the intersection of the rectangle and the parallelogram. The feasible region is shown in figure 3.7 as the intersection of the rate and the amplitude constraints. As mentioned in the previous subsection, the size of the parallelogram may vary at each sampling instant. Hence, unlike the rate constraints or the amplitude constraints, the geometry of the rate plus amplitude constraints is not fixed. The geometry of the feasible region in this case can be a polygon of three to six sides.

3.2.3 Geometric Interpretation of the Optimal Solution

In the previous sections 3.2.1 and 3.2.2, the geometric shapes of the cost-function of GPC and the feasible regions of the rate, amplitude and rate plus amplitude constraints were illustrated. In this section, a geometric explanation of the optimal solution to the constrained cost-function will be presented.

The objective of an unconstrained cost-function is to find a $\tilde{\mathbf{u}}^*$ such that the value of J_{GPC} in equation 3.10 is minimum. Geometrically, the objective is to find a $\tilde{\mathbf{u}}^*$ such

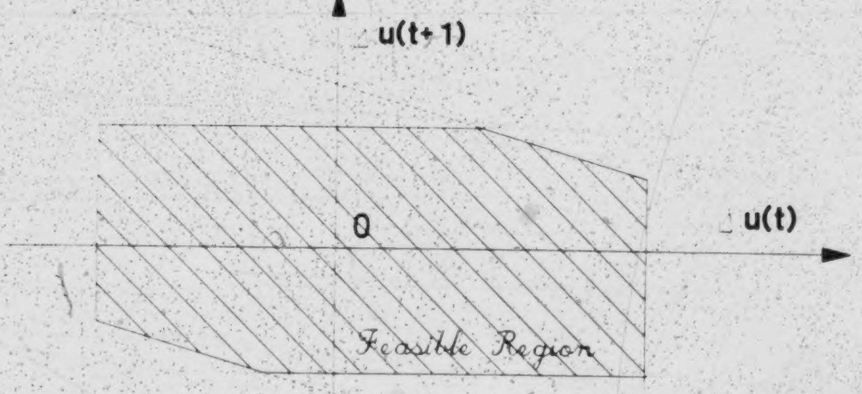
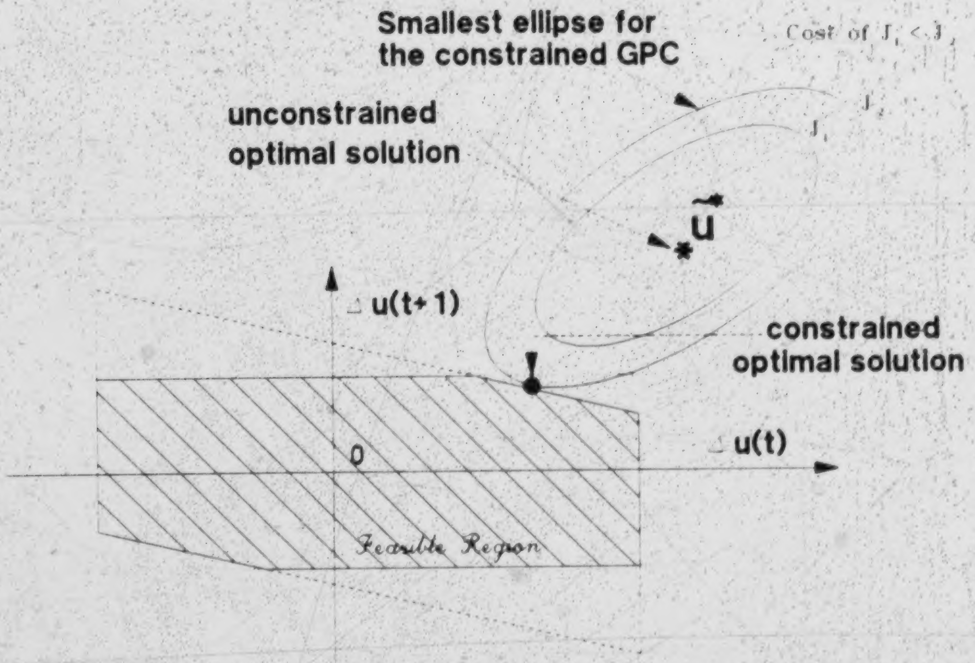


Figure 3.7: Feasible region of the rate plus amplitude constraints

that the ellipse is of smallest size. Geometrically, when the cost function is subjected to constraints, the objective is to find a solution $\bar{\mathbf{u}}^{+}$ such that the ellipse is of smallest size and *the solution also belongs to the feasible region*. This is illustrated by the figure 3.8. The cost of ellipse J_1 is less than the cost of ellipse J_2. $\bar{\mathbf{u}}^{*}$ is a point ellipse and has the smallest possible cost. The J_2 ellipse is the smallest ellipse which has a point $\bar{\mathbf{u}}^{+}$ in common with the feasible region. The optimal solution of the unconstrained cost function is $\bar{\mathbf{u}}^{*}$ and of the constrained cost function is $\bar{\mathbf{u}}^{+}$.

If the unconstrained optimal solution lies in the feasible region, then it is also the optimal solution for the constrained problem. However, if the unconstrained solution lies outside the feasible region then the optimal solution is on the boundary of the feasible region. This property of the optimal solution to the constrained cost functions has been used in developing the analytical solution in section 3.1.



Smallest ellipse for the constrained GPC
Cost of $J_1 < J_2$
unconstrained optimal solution
J_2
J_1
$\tilde{u}^*$
$\Delta u(t+1)$
constrained optimal solution
$\Delta u(t)$
0
Feasible Region

Figure 3.8: Optimal solution of a constrained cost function

3.3 Conclusions

Analytical solutions of constrained SISO GPC have been developed for $NU = 1$ and 2, and for MGPC for $NU = 1$. It is computationally much more efficient than algorithmic solution of constrained GPC and guarantees convergence. For most of the applications, a control horizon setting of $NU = 1$ or 2 will result in a reasonably good control and hence the algorithm is of practical interest.

GPC cost function can be geometrically interpreted as an equation to loci of ellipses, for $NU = 2$, on the $\Delta u(t)$—$\Delta u(t+1)$ plane. The rate constraints form a rectangle and the amplitude constraints form a parallelogram on the $\Delta u(t)$—$\Delta u(t+1)$ plane. The rate plus amplitude constraints are represented by the intersection of the rectangle and the parallelogram and can have a geometry of a three sided to six sided polygon, on the $\Delta u(t)$—$\Delta u(t+1)$ plane. If the solution of the unconstrained GPC does not lie in the feasible region, then the solution of the constrained GPC is the point of contact of the smallest ellipse touching the feasible region, on the $\Delta u(t)$—$\Delta u(t+1)$ plane.

Chapter 4

Performance Evaluation of the MGPC

In the previous chapters, a long range predictive control algorithm—multivariable generalized predictive control (MGPC) was presented and an analytical solution for some special cases of the constrained control algorithm were developed. In this chapter the MGPC algorithm will be evaluated for its performance on a simulated process. The performance of a multivariable control scheme can be evaluated in terms of its performance in handling interactions among different loops, disturbance rejection properties–with and without feedforward compensation, and model process mismatch. In this chapter, the performance of multivariable generalized predictive control (MGPC) is evaluated for all these criterions, by simulating a discretized 3 × 3 multivariable process.

The organization of this chapter is as follows: Section 4.1 describes the software developed for implementing the MGPC; section 4.2 considers the analysis and discussion of the Shell heavy oil fractionator problem [42] for interactions among different loops, disturbance rejection properties and model process mismatch; followed by concluding remarks in section 4.3.

4.1 Software Development for MGPC

Chapter 3 describes the analytical solution for rate and amplitude constrained GPC, for some special cases of controller settings. Though it is computationally efficient, it cannot handle all possible setting of the controller as elegantly and efficiently. Therefore, an software package was developed which uses algorithmic solution to implement constrained GPC algorithm. The MGPC algorithm described in Chapter 2 was programed in 'C' and FORTRAN for evaluating the algorithm by simulating examples and conducting experimental runs.

The developed software uses a quadratic programming (QP) optimization package for solving the constrained cost function generated by the MGPC. QPSOL, a QP package developed at the Stanford University, Stanford, by Gill *et al.* [19], and TOLMIN, a linearly constrained optimization calculations package developed at the University of Cambridge, by Powell [41], were considered for solving the linearly constrained quadratic cost function of MGPC. QPSOL and TOLMIN are briefly described as follows:

QPSOL: is a set of FORTRAN subroutines designed to locate the minimum value of an arbitrary quadratic function subject to linear constraints and simple upper and lower bounds. The problem is assumed to be stated in the following form:

$$QP \qquad \text{minimize} \quad c^T x + \frac{1}{2} x^T H x$$

$$\text{subject to} \quad l \le \left\{ \begin{array}{c} x \\ \\ Ax \end{array} \right\} \le u$$

where c is a constant n-vector and H is a constant $n \times n$ symmetric matrix; note that H is the Hessian matrix (matrix of second partial derivatives) of the quadratic objective function. The matrix A is $m \times n$ where m may be zero; A is treated as a dense matrix. The algorithm is not intended for large sparse problems.

If the quadratic function is convex, a global minimum is found; otherwise, a local minimum is found. QPSOL is most efficient when many constraints or bounds are active at the solution. The algorithm is good at handling problems with up to several hundred general constraints and relatively few variables, or several hundred variables and relatively few general constraints. This program can also be used for solving linear programming (LP) problems by setting the Hessian H to zero matrix or by setting a flag in the software package to solve LP problems.

QPSOl requires approximately 75 kilobytes of storage for program source code, besides workspace. The package contains approximately 6000 lines of ANSI (1966) standard FORTRAN, of which 44% are comments. It can be implemented on-line for real-time applications, as demonstrated by the experimental runs. The execution time is of the order of hundreds of milliseconds for a QP problem with 6 to 9 variables subjected to 60 to 80 constraints on a IBM PS/2 model 70 386 computer.

TOLMIN: is a package of FORTRAN subroutines for the minimization of a differentiable function $F(x)$ of N variables subject to linear constraints. The problem solved by TOLMIN can be stated as follows:

$$\begin{aligned} &\text{minimize} && F(x) && \\ &\text{subject to} && a_j^T x = b_j && j = 1, 2, \ldots, MEQ \\ & && a_j^T x \leq b_j && j = MEQ + 1, \ldots, M \\ & && l \leq x \leq u && \end{aligned}$$

on the vector of variables x. This method is particularly suitable if there are any degenerate or nearly-degenerate constraints, because no search is allowed to move towards the boundary of a constraint that has a small residual at the initial point of a line search. Otherwise it is a typical active set method (Gill *et al.* [18]).

TOLMIN contains approximately 1500 lines of source code. The code does not use any two-dimensional arrays or private working space in order that there are no upper bounds on the numbers of variables or constraints and for achieving better computational efficiency. This package also can be implemented on-line for real-time applications.

As can be seen from the objective functions of both the packages, TOLMIN solves more general problems than QPSOL. Also, the source code of TOLMIN is smaller than that of QPSOL. However, it was observed for several runs that the convergence properties of QPSOL were better than that of TOLMIN for quadratic cost functions subjected to constraints. As GPC is based on a quadratic cost function, QPSOL was used for minimizing the cost-function in the software developed for implementing MGPC.

The convergence properties of QPSOL were reasonably good for most of the simulations and experimental runs conducted with this software. It can handle rate constraints very well and amplitude constraints reasonably well. The handling of the output constraints is not very good but still manageable.

The alternative of using a cost function minimizing the sum of absolute errors results in a linear programming (LP) problem. The computational load in solving a LP problem is much lower than solving a QP problem. However, the use of LP cost function results in relatively poor dynamic decoupling than QP cost function. This is due to the fact that LP will always results in a solution at the tip of constraints whereas QP can also have a solution in the feasible region or at the boundaries (*not necessarily tips*) of constraints. Depending on the requirements of a process, the GPC cost function can be selected as a LP or QP problem.

Programming language 'C' was used for the control algorithm because the algorithm was implemented on the experimental process under the operating system QNX [61], which is marketed with a 'C' compiler. Secondly, the versatile nature and functionality of 'C' makes it the obvious choice. It is possible to dynamically allocate memory in 'C' and therefore, its use results in efficient use of computer memory. The functional nature of 'C' makes this programming language convenient for writing various subsets of application programs by different people and improves portability of the different control algorithms. As mentioned earlier, QPSOL is available in FORTRAN. Therefore, to solve the objective function generated by the control algorithm, mixed language programming was necessary for implementing the software package. The software package developed has some nice features:

- It can implement MGPC algorithm for an n-output m-input system, where n is greater than, equal to or less than m.

- It can provide *dynamic* feedforward compensation.

- If a multivariable identification package is available, the software package can be implemented as an adaptive control algorithm. This can be achieved by setting a flag as ON or OFF for identification (in the MULTICON data table for experimental runs). If the flag is set to ON, then one of the functions in the control algorithm will read the new parameters of the process and update all the relevant variables for the new set of parameters. In this thesis, only SISO experimental runs were implemented as an adaptive control strategy.

- The memory is used more efficiently because of dynamic memory allocation in the control algorithm.

4.1.1 The MGPC Algorithm

The listing of the programs developed for simulations and experimental runs is presented in appendix C. The MGPC algorithm is implemented as follows:

- Set MGPC tuning parameters N_1, N_2 NU and all the flags to ON or OFF. The different flags and their status will be mentioned at the appropriate place in the algorithm.

- Create all matrices and vectors of size null.

- Open files for data acquisition.

- Read the process transfer function matrix description from a function.

- Read the setpoints, lower and upper rate and amplitude constraints on manipulated variables and lower and upper constraints on process output.

```
• Calculate the carima model from the transfer function matrix description and redimension all matrices and vectors to appropriate dimensions as required by the process description. Also, generate the step response matrix $\mathcal{G}$.

• Generate the constraint matrix as required by QPSOL.

• Vectorize $\Lambda$, upper and lower bounds of constraints depending on $N_1$, $N_2$, $NU$ and number of inputs and outputs of the process.

• Generate the Hessian matrix of the cost function.

• do: {

  I if the flag for the algorithm to be on/off is non-zero then break from do loop.

  II if the flag for controller to be on/off is non-zero then MGPC is ON {

      1. if the flag for process identification to be on/off is non-zero then the process parameter update is ON {

         a. Update process transfer function matrix.

         b. Generate carima model from the updated transfer function matrix.

         c. Set the flag new_theta to ON.

         }

         else set the flag new_theta to OFF.

      2. Read new controller parameters ($N_1$, $N_2$, $NU$, $\Lambda$, and constraints) and process setpoint. If any of these values have changed or if the flag new_theta is ON then recalculate the affected variables among the step response matrix $\mathcal{G}$, Hessian matrix, vectorized lower and upper bounds of the constraints, the constraint matrix, vectorized setpoint vector.

      3. Calculate GPC control action {

         a. Calculate free-response of the process.
```

```
            b. Setup the Hessian matrix, free-response vector, constraint matrix, and
               lower and upper bounds on the constraints as required by QPSOL.
            c. Copy all matrices (with dynamic memory allocation) to two dimensional
               arrays for achieving FORTRAN and 'C' compatibility during transfer of
               variables between them.
            d. Invoke the QPSOL program (which is in FORTRAN) to solve the
               quadratic programming problem generated by MGPC for calculating
               the control action.
            }
        } else Controller is OFF. Read the user specified control signal to be implemented.
    III Implement the control action.
    IV Wait for next sampling interval.
    V Read the Process Output. The implementing of control signal and reading of
        process output for experimental runs is done by a program which interfaces the
        process and the control algorithm. For simulations, the output of the process is
        calculated for the implemented inputs.
    VI if data acquisition flag is ON, then write the implemented control actions and
        the process outputs to a file.
    VII Update the appropriate input and output data vectors and matrices.
    }
• Free all memory allocated to the vectors and matrices. Close all the data acquisition
  files.
• end
```

4.2 The Shell Control Problem–A Case Study

To evaluate the performance of MGPC, the control of the Shell heavy oil fractionator as documented in the Shell Process control Workshop [42] was simulated for different criterions listed in the introduction of this chapter.

This section evaluates the performance of the fixed parameters MGPC on the Shell control problem. The Shell control problem is first described in section 4.2.1; section 4.2.2 illustrates the decoupling achieved by MGPC; section 4.2.3 evaluates the load rejection properties of MGPC with and without feedforward compensation; and section 4.2.4 evaluates the performance of MGPC in the presence of model-process mismatch.

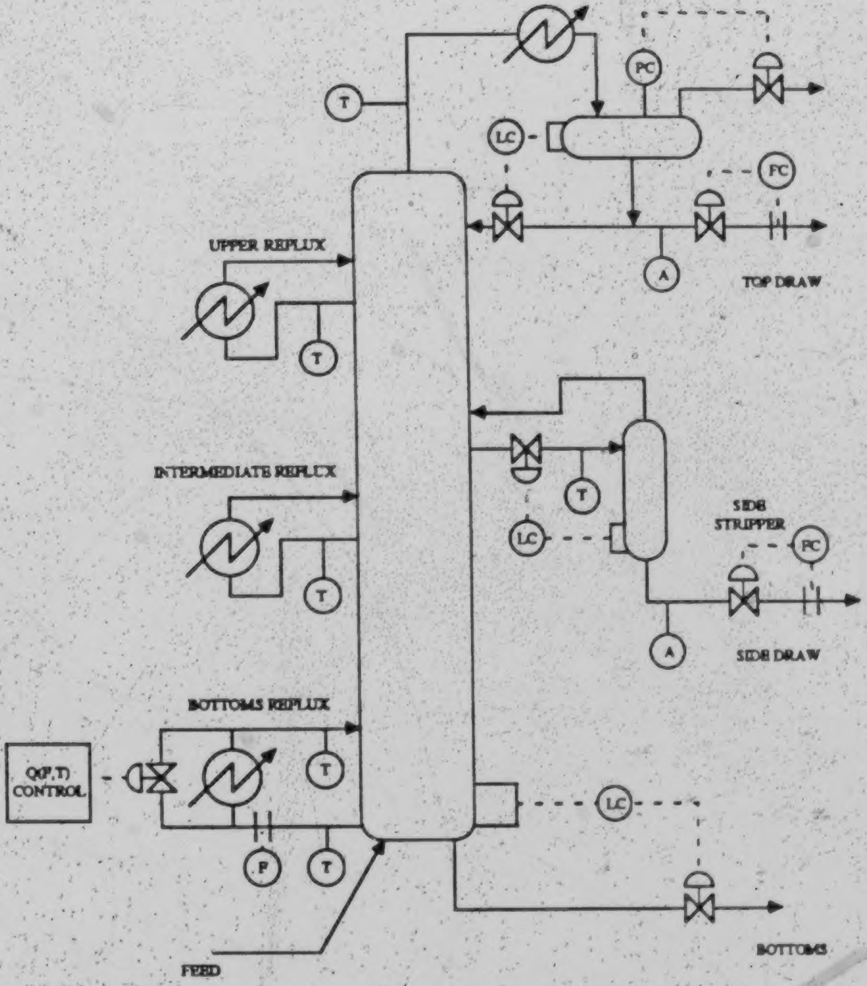
4.2.1 The Shell Control Problem

Figure 4.1 shows the heavy oil fractionator used as the Shell control problem [42], with three product draws and three side circulating loops. The fractionator can be described as follows: The heat requirement of the column enters with the feed, which is a gaseous stream. Product specifications for the top and side draws are determined by economics and operating requirements. There is no product specification for the bottom draw, but there is an operating constraint on the temperature in the lower part of the column. The three circulating loops in these loops reboil columns in other parts of the plant. Therefore, they have varying heat duty requirements. The bottom loop has an enthalpy control which regulates heat removal in the loop by adjusting steam make. Its heat duty can be used as a manipulated variable to control the column. The heat duties of the other two loops act as disturbances to the column.

The relevant information regarding the Shell control problem is stated in the following two subsections.

4.2.1.1 Control Objectives and Constraints

The *control objective* of the simulations can be stated as follows:



T
PC
LC
FC
UPPER REFLUX
T
A
TOP DRAW
INTERMEDIATE REFLUX
T
LC
SIDE STRIPPER
FC
T
A
SIDE DRAW
BOTTOMS REFLUX
Q(F,T) CONTROL
T
P
T
LC
FEED
BOTTOMS

Figure 4.1: Schematic diagram of the heavy oil fractionator used as the Shell control problem

1. Maintain the top and side-draw product end points at the specifications. The original problem [42] specifies regulatory control (0.0 ± 0.005 at steady state). But to illustrate the performance more clearly, all the simulations were performed with servo-control strategy, with the set-points being ± 0.05.
2. Reject the unmeasured disturbances entering the column from the upper and intermediate refluxes due to change in heat duty requirements from other columns.

The above control objectives were subjected to the following *constraints*:

1. All draws must be within hard maximum and minimum bounds of 0.5 and -0.5.
2. The bottom reflux heat duty is also constrained within the hard bounds of 0.5 and -0.5.
3. All manipulated variables have maximum move size limitations of magnitude 0.05 per minute.
4. Fastest sampling time is 1 minute.
5. The bottom reflux draw temperature has a minimum value of -0.5.
6. The top end point must be maintained within the maximum and minimum values 0.5 and -0.5.

4.2.1.2 Process Model and Uncertainties in the Gains of the Model

The Shell control problem describes a 7-output 5-input process to model the heavy oil fractionator shown in figure 4.1. For evaluation purposes, it is easier to run a smaller dimensional problem since they are more tractable. The original 7 × 5 problem does indeed run on the software. However, for the ease of evaluation and understanding, a smaller order problem of 3-outputs and 5-inputs is considered here.

For the simulations, a subset of 3-output 3-input process was selected as the main process and the remaining two inputs were used as disturbance variables. The control algorithm was implemented subject to all the constraints described in section 4.2.1.1, for all the simulations performed on the fractionator. The process model is described by a collection of first order plus time delay transfer function relating the input-output pairs. The parameters of the transfer function represented as $\frac{Ke^{-\theta s}}{\tau s+1}$ are given in table 4.1.

The three outputs of the simulated system are top end point, side end point, and bottom reflux temperature and the three inputs are top draw, side draw, and bottoms reflux duty. Intermediate reflux duty and upper reflux duty are the disturbance variables. The model was discretized with a sampling time of 4 minutes. This is compatible with most of the time constants. This choice was also necessitated to avoid large sample interval delays. The open loop step response time-trajectory plot for this system is shown in figure 4.2. The tower exhibits significant interaction for all the inputs.

To evaluate the performance of MGPC in handling model-process mismatch, the steady state gain mismatch criterion suggested in Shell control problem [42] were used and are reproduced in table 4.2 for convenience.

The Shell control problem defined in the above sections was simulated to evaluate the performance of the constrained MGPC algorithm and is described in the following subsections. All the simulations were performed with the model as defined in table 4.1 and discretized with a sampling time of 4 minutes. All the simulations were subjected to the constraints defined in section 4.2.1.1. Amplitude constraints were set up as suggested in the Shell control problem. The $T(q^{-1})$ polynomial was set equal to identity, unless otherwise defined, for all the simulations. This was partly due to the fact that exact dynamics of the process were used for all the simulations and partly for isolating the effects of various tuning parameters. All the figures for the simulations are presented as outputs versus sampling instances (which are 4 minutes each).

	TOP DRAW	SIDE DRAW	BOTTOMS REFLUX DUTY	INTER'M REFLUX DUTY	UPPER REFLUX DUTY
	u_1	u_2	u_3	d_1	d_2
TOP	$K = 4.05$	$K = 1.77$	$K = 5.88$	$K = 1.22$	$K = 1.44$
END y_1	$\tau = 50$	$\tau = 60$	$\tau = 50$	$\tau = 45$	$\tau = 40$
POINT	$\theta = 27$	$\theta = 28$	$\theta = 27$	$\theta = 27$	$\theta = 27$
SIDE	$K = 5.39$	$K = 5.72$	$K = 6.90$	$K = 1.52$	$K = 1.83$
END y_2	$\tau = 50$	$\tau = 60$	$\tau = 40$	$\tau = 25$	$\tau = 20$
POINT	$\theta = 18$	$\theta = 14$	$\theta = 15$	$\theta = 15$	$\theta = 15$
BOTTOMS	$K = 4.38$	$K = 4.42$	$K = 7.20$	$K = 1.14$	$K = 1.26$
REFLUX y_3	$\tau = 33$	$\tau = 44$	$\tau = 19$	$\tau = 27$	$\tau = 32$
TEMP	$\theta = 20$	$\theta = 22$	$\theta = 0$	$\theta = 0$	$\theta = 0$

Table 4.1: The process model of transfer function parameters

	u_1	u_2	u_3	d_1	d_2
y_1	$4.05 + 2.11\epsilon_1$	$1.77 + 0.39\epsilon_2$	$5.88 + .59\epsilon_3$	$1.22 + 0.12\epsilon_4$	$1.44 + 0.16\epsilon_5$
y_2	$5.39 + 3.29\epsilon_1$	$5.72 + 0.57\epsilon_2$	$6.90 + 0.89\epsilon_3$	$1.52 + 0.13\epsilon_4$	$1.83 + 0.13\epsilon_5$
y_3	$4.38 + 3.11\epsilon_1$	$4.42 + 0.73\epsilon_2$	$7.20 + 1.33\epsilon_3$	$1.14 + 0.18\epsilon_4$	$1.26 + 0.18\epsilon_5$

$$-1 \leq \epsilon_i \leq 1; \quad i = 1, 2, 3, 4, 5$$

Table 4.2: The suggested model-process gains mismatch in the Shell control problem

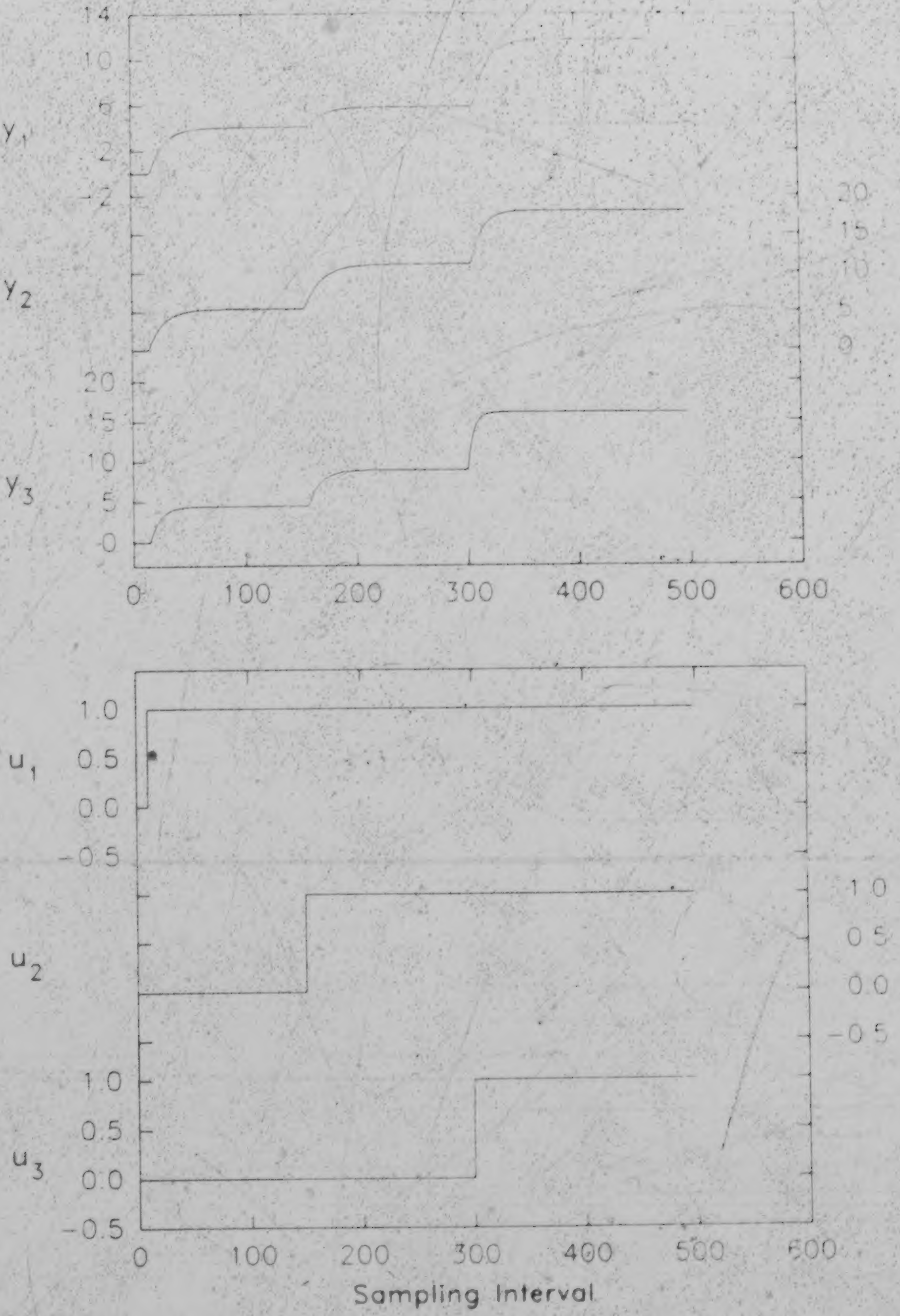


Figure 4.2: Open loop step response characteristics of the Shell control problem

4.2.2 Interaction Between Different Outputs of the Closed Loop System

This section shows the amount of interaction between the different outputs of the closed loop system. The simulations were performed with the constraints specified in section 4.2.1.1 and with different values of rate constraints, specified in next paragraph.

The simulations in figures 4.3 and 4.4 were performed with MGPC tuning knobs set as $N_1 = 1$, $N_2 = 15$, $NU = 2$, $\lambda_1 = \lambda_2 = \lambda_3 = .01$, and $y_{wt1} = y_{wt2} = y_{wt3} = 1$. The rate constraints were $\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{max} = 0.2$ and $\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{min} = -0.2$ for the simulation in figure 4.3 and $\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{max} = 0.0025$ and $\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{min} = -0.0025$ for the simulation in figure 4.4.

From the time trajectory plots shown in figures 4.3 and 4.4, it is clear that the closed loop MGPC significantly reduces the interactions between different outputs when compared with the figure 4.2, which shows the extent of interaction present in the open-loop plant. Also, as can be seen from figure 4.4, when the rate constraints are narrowed down to a smaller value, the control is sluggish and the decoupling is 'smoother'. Fast or rapid control action is not desirable. Hence the performance of the MGPC with the setting for figure 4.4 is better than that of figure 4.3. It must be mentioned that for $NU = 1$ and large N_2, the MGPC control algorithm would perform like a steady state controller. Increasing the value of NU to values greater than 2 will result in better decoupling of different outputs at the cost of more aggressive control action.

It is possible to decrease the interaction between the different outputs by scaling the outputs (with y_{wt}), as suggested by Maurath *et al.* [34]. Decoupling of a particular loop may be enhanced by weighting the corresponding output significantly more than the others, thus decoupling that output from set-point changes in other loops. Also, the decoupling can be enhanced by using a particular time-varying set-point trajectory to enforce decoupled set-point changes in the closed loop system (Maurath *et al.* [34]).

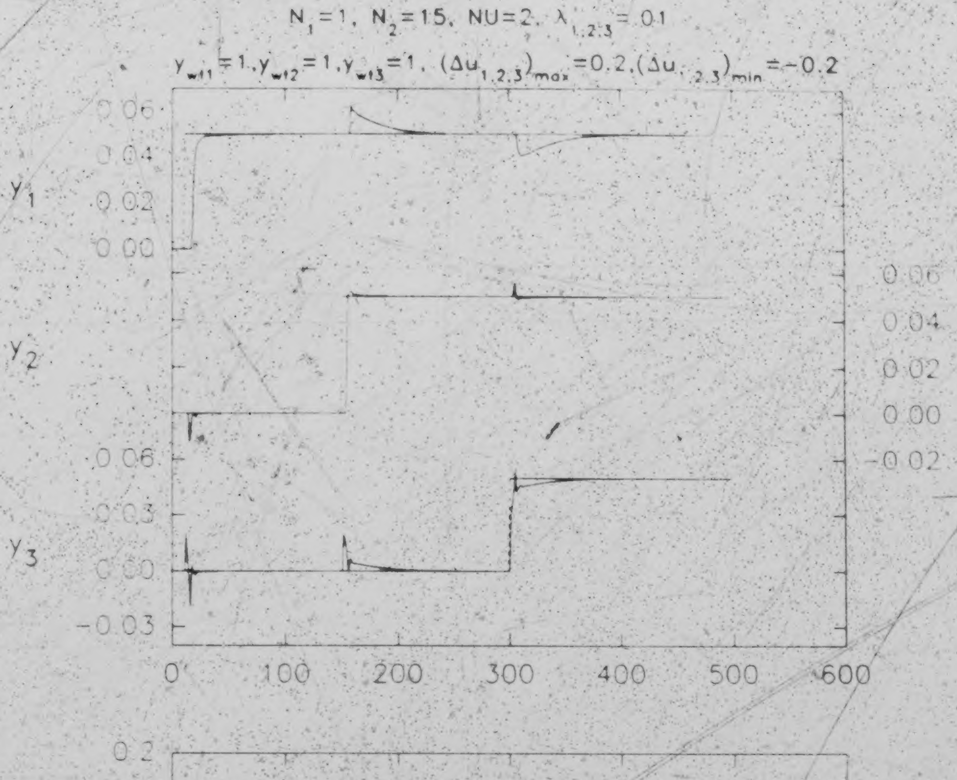


Figure 4.3: Interaction between different outputs of the closed loop system in the Shell control problem with rate constraints of ±0.2

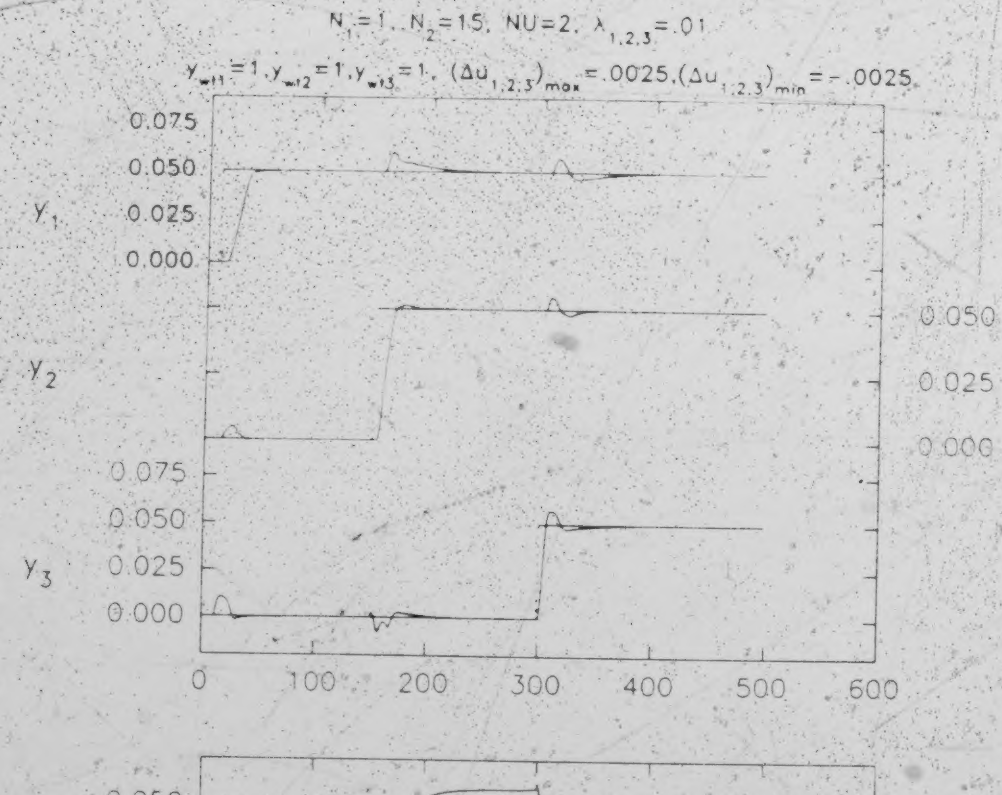


Figure 4.4: Interaction between different outputs of the closed loop system in the Shell control problem with rate constraints of ±0.0025

4.2.3 Load Rejection, with and without Feedforward Compensation

This section evaluates the performance of the MGPC algorithm for load rejection properties for the Shell control problem. All the simulations were performed with the constraints specified in 4.2.1.1. The variables intermediate-reflux duty (d_1) and upper-reflux (d_2) duty in table 4.1 were used as the disturbance variables and were simultaneously given step changes. Simulations were performed with and without feedforward compensation.

4.2.3.1 Without Feedforward Compensation

The simulations in figures 4.5 and 4.6 were performed with MGPC tuning knobs set as $N_1 = 1$, $N_2 = 15$, $\lambda_1 = \lambda_2 = \lambda_3 = .01$, and $y_{wt1} = y_{wt2} = y_{wt3} = 1$, $\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{max} = 0.2$ and $\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{min} = -0.2$. The control horizon, NU, for figure 4.5 was 1 and for simulation in figure 4.6 was 2. The disturbance variables d_1 and d_2 were simultaneously given step changes between $+0.002$ and -0.002.

From the time-trajectory plots of the simulations it is observed that the controller performance with $NU = 1$ (figure 4.5) is better than with $NU = 2$ (figure 4.6). With $NU = 2$, the control action is more aggressive and the load rejection is relatively poor and hence is not desirable. Though the controller successfully rejects the load disturbances, it is possible to enhance the performance of the system, if the load disturbances d_1 and/or d_2 are measurable, by incorporating feedforward compensation in the control algorithm.

4.2.3.2 With Feedforward Compensation

The performance of MGPC improves significantly when feedforward compensation for measurable disturbances is included in the control algorithm. The implementation of MGPC algorithm for *dynamic* feedforward compensation is described in section 2.4.2.

The simulations in figures 4.7, 4.8 and 4.9 were performed with MGPC tuning knobs set as $N_1 = 1$, $N_2 = 15$, $\lambda_1 = \lambda_2 = \lambda_3 = .01$, and $y_{wt1} = y_{wt2} = y_{wt3} = 1$,

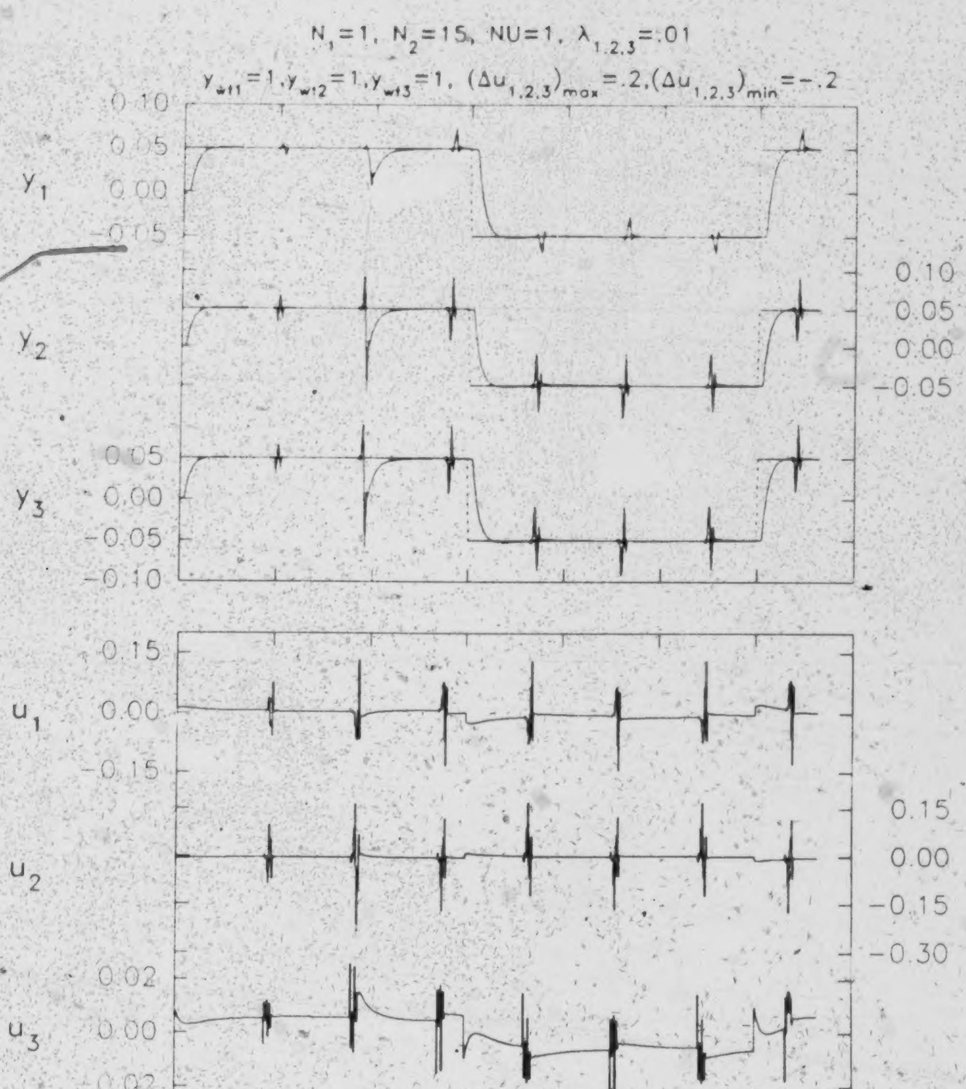


Figure 4.5: Load rejection by MGPC without feedforward compensation for the Shell control problem with $NU = 1$

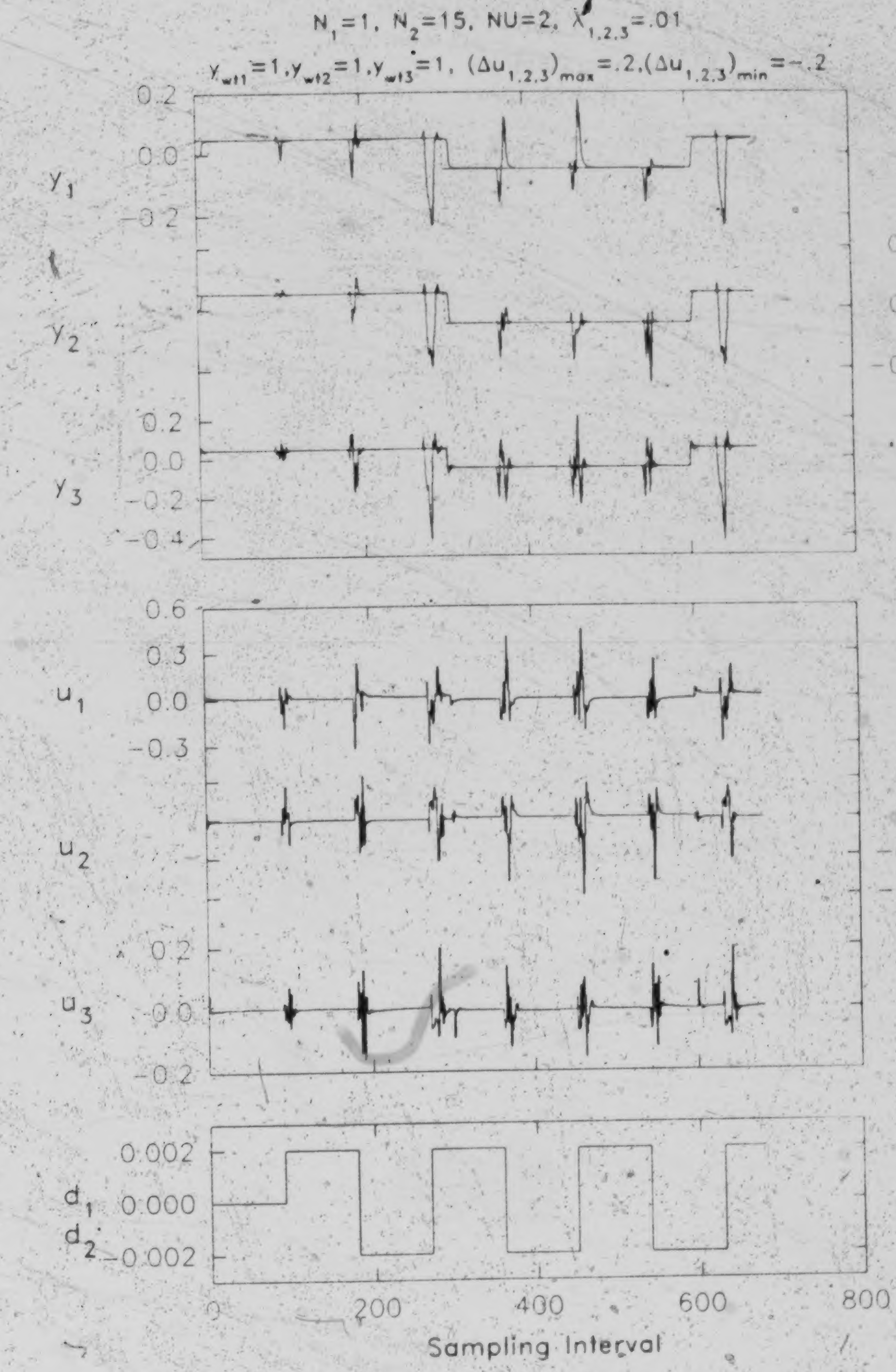


Figure 4.6: Load rejection by MGPC without feedforward compensation for the Shell control problem with $NU = 2$

$\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{max} = 0.2$ and $\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{min} = -0.2$. The control horizon, NU, for simulation in figure 4.7 was 1, in figure 4.8 was 2, and in figure 4.9 was 3. The disturbance variables d_1 and d_2 were simultaneously given step changes between +0.01 and −0.01, five times larger than in the simulations without feedforward compensation.

The time-trajectory plots (figures 4.7, 4.8, 4.9) of these simulations show that the performance of the feedforward compensated system is much better than the performance of the systems without feedforward compensation (figures 4.5 and 4.6). It is also seen that the performance of the feedforward compensated system improves with increase in control horizon, NU, unlike the performance of the system without feedforward compensation.

4.2.4 Effect of Model Process Mismatch

This section evaluates the effect of model plant mismatch (**MPM**) on the performance of the Shell control problem with MGPC algorithm. While MPM is always a reality, the extent and type of mismatch that commonly occur can differ in different processes. Variable delays and time-constants are common types of MPM problems. Variation in steady-state gain, on the other hand is relatively easy to monitor and can be stored in the computer to be employed on a table-look-up basis. However, for the the purpose of this simulation, GPC control in the presence of MPM in the steady-state gains was evaluated as if an abrupt change in the steady-state gain does occur as *suggested* in the Shell control problem [42]. A control algorithm will not be of practical importance if it cannot handle MPM. The MPM at high frequencies is due to unmeasured dynamics and noise and at low frequencies is due to steady state gain mismatch. The high frequency mismatch can be handled with the help of the $T(q^{-1})$ polynomial (section 2.4.1). Mohatadi [36] demonstrates the necessity of the $T(q^{-1})$ polynomial in handling unmodelled dynamics. The steady state gain mismatch of the process and the model can be handled by the integral action, from the CARIMA model, of the controller.

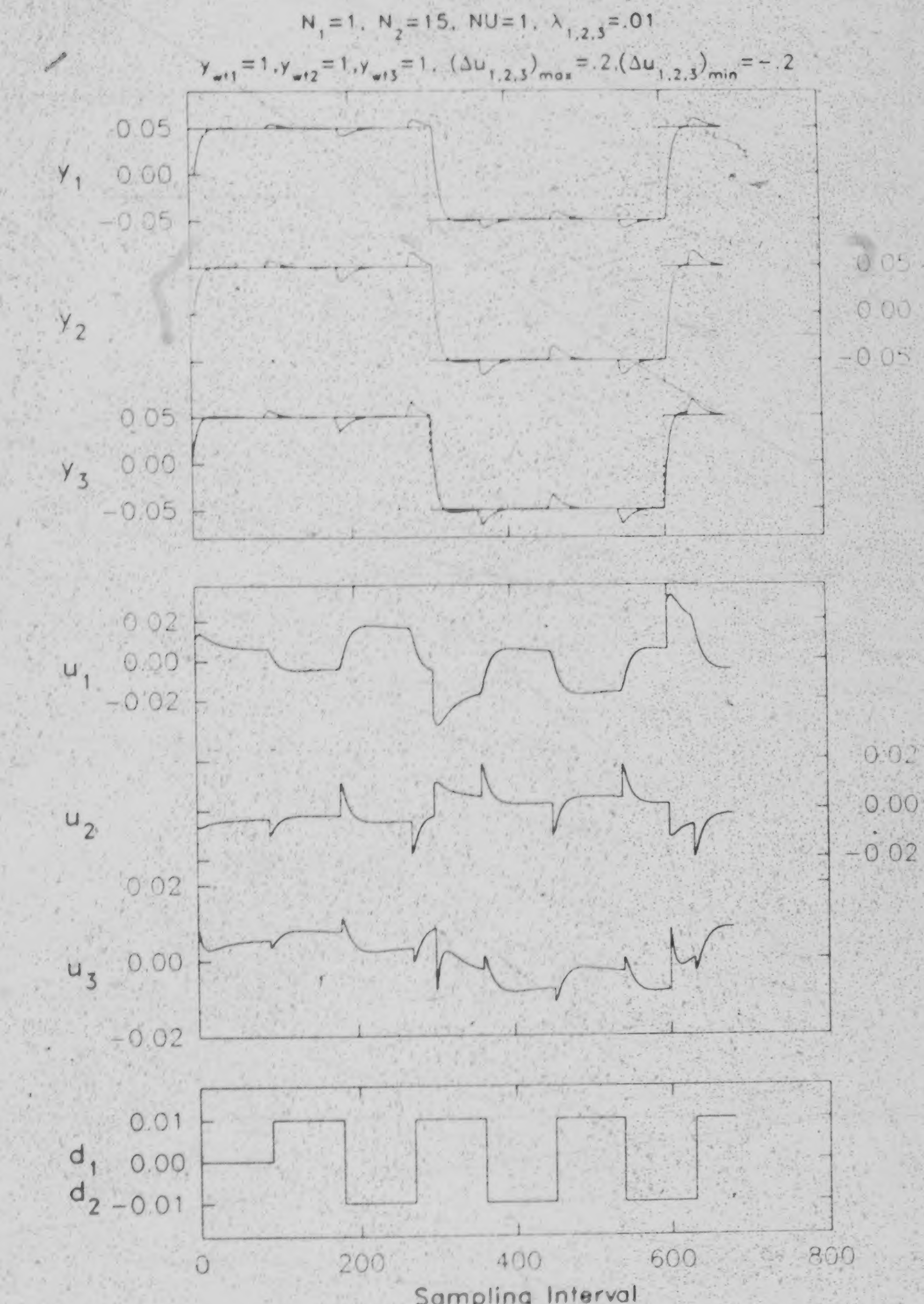


Figure 4.7: Load rejection by MGPC with feedforward compensation for the Shell control problem with $NU = 1$

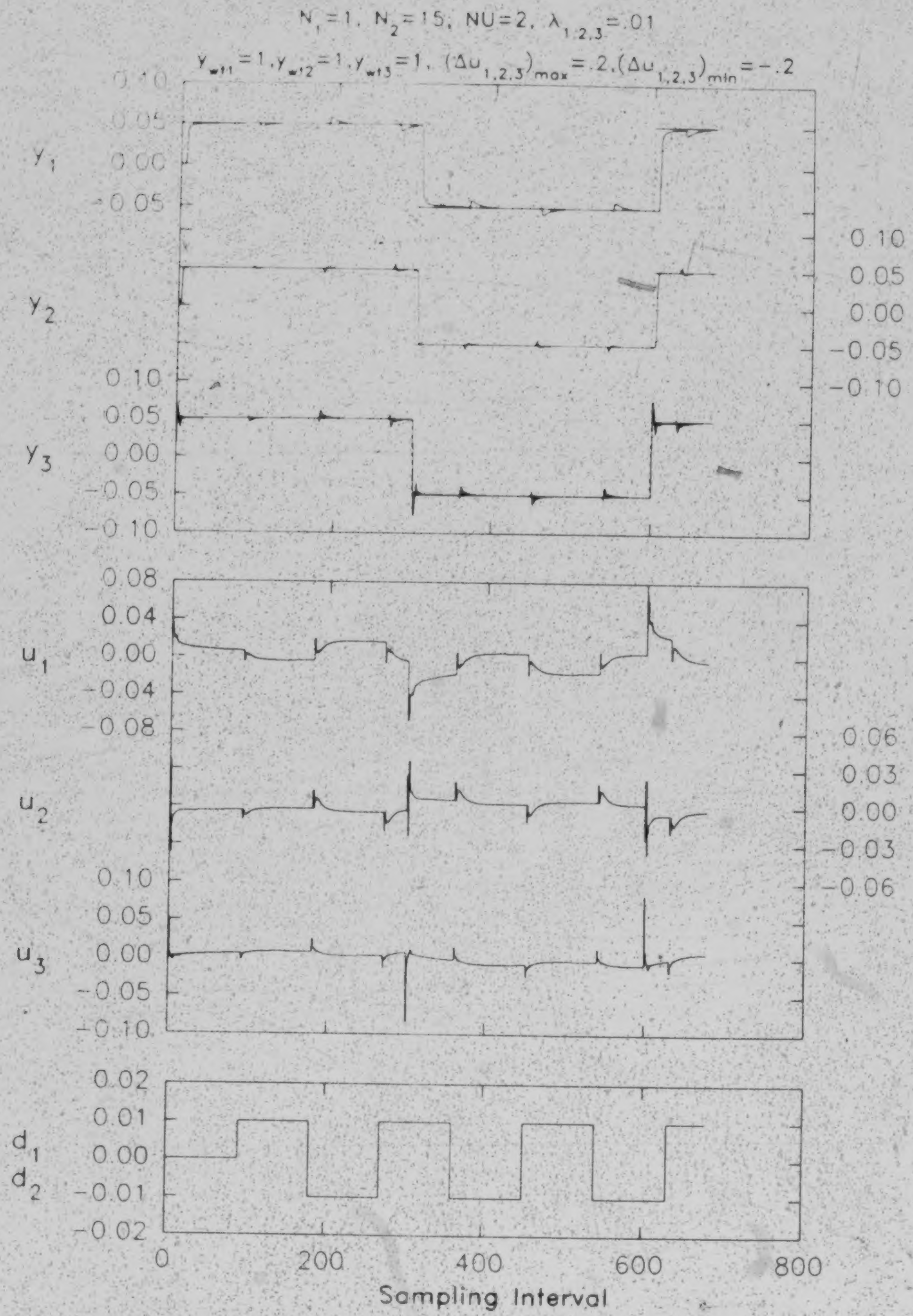


Figure 4.8: Load rejection by MGPC with feedforward compensation for the Shell control problem with $NU = 2$

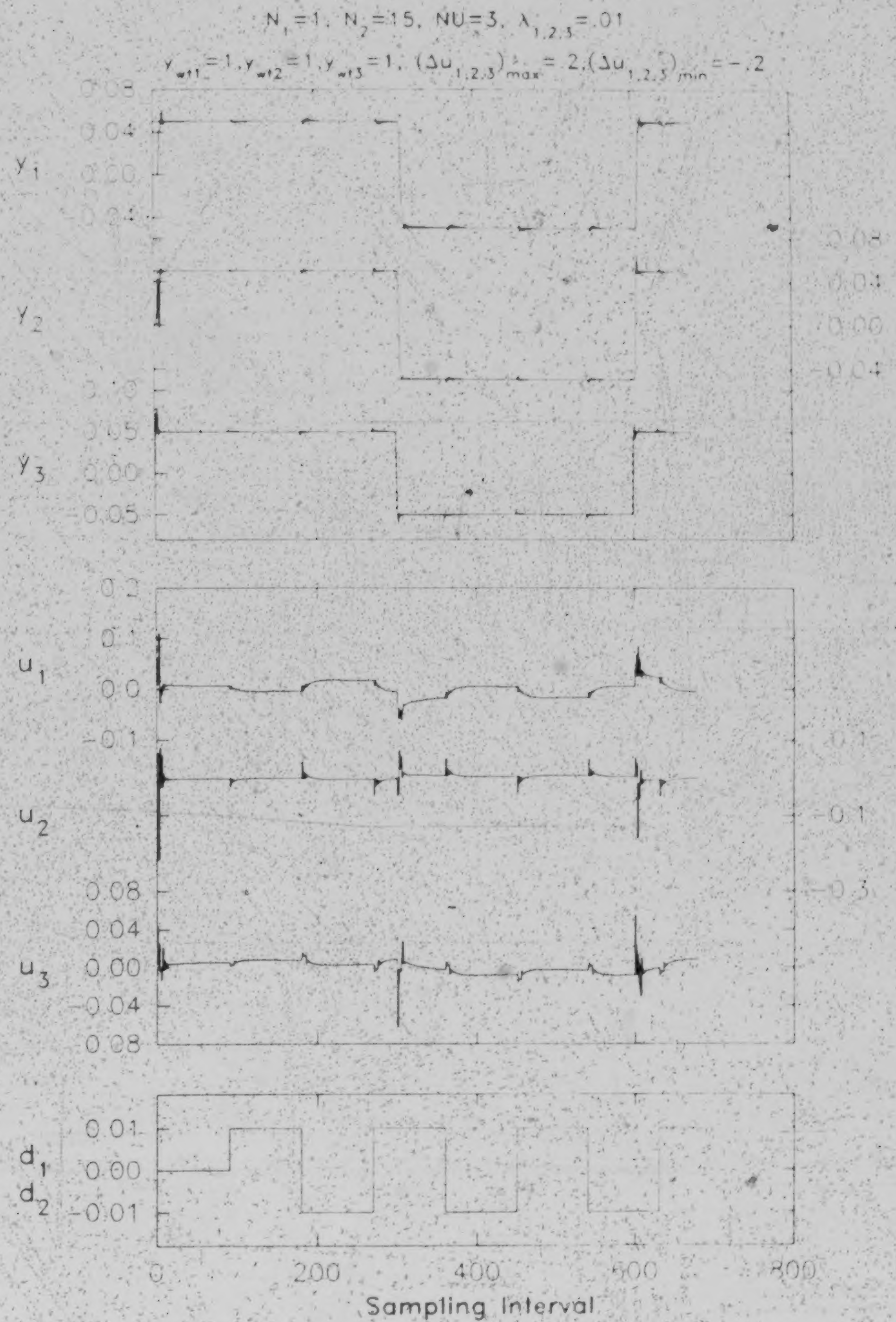


Figure 4.9: Load rejection by MGPC with feedforward compensation for the Shell control problem with $NU = 3$

In this section, a thorough evaluation of the MGPC algorithm for low frequency MPM is presented with the help of the Shell control problem simulations. The effect of different tuning parameters of MGPC on a system with MPM is also illustrated through these simulations. The gains for mismatch were selected from the table 4.2. All the simulations were performed with constraints specified in section 4.2.1.1. In the following subsection, the controller settings of the simulations are described first followed by a brief description of the performance. Discussion on these simulations is also presented under separate headings for different tuning parameters.

Note: **Notation:**

G_{ij} is the (i,j) element of the transfer function matrix *i.e.* i_{th} output j_{th} input channel of the process.

4.2.4.1 MPM in G_{11}:

The model process mismatch (MPM) was introduced in channel G_{11} by changing the steady state gain of the process to 6.16 from 4.05. The simulation in figure 4.10 was performed with MGPC tuning knobs set as $N_1 = 1$, $N_2 = 15$, $\lambda_1 = \lambda_2 = 15$, $\lambda_3 = 25$, and $y_{wt1} = 25$, $y_{wt2} = y_{wt3} = 1$, $\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{max} = 0.001$ and $\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{min} = -0.001$. The control horizon, NU, for the simulation was 1 and the MPM was introduced at the sampling interval 400.

From the time-trajectory plot in figure 4.10 it is observed that the control algorithm was able to cope up with the model process mismatch in channel G_{11}. The reason for selecting the tuning parameter y_{wt} to a value of 25 for channel G_{11} will be clear from the discussion on effect of output weighting.

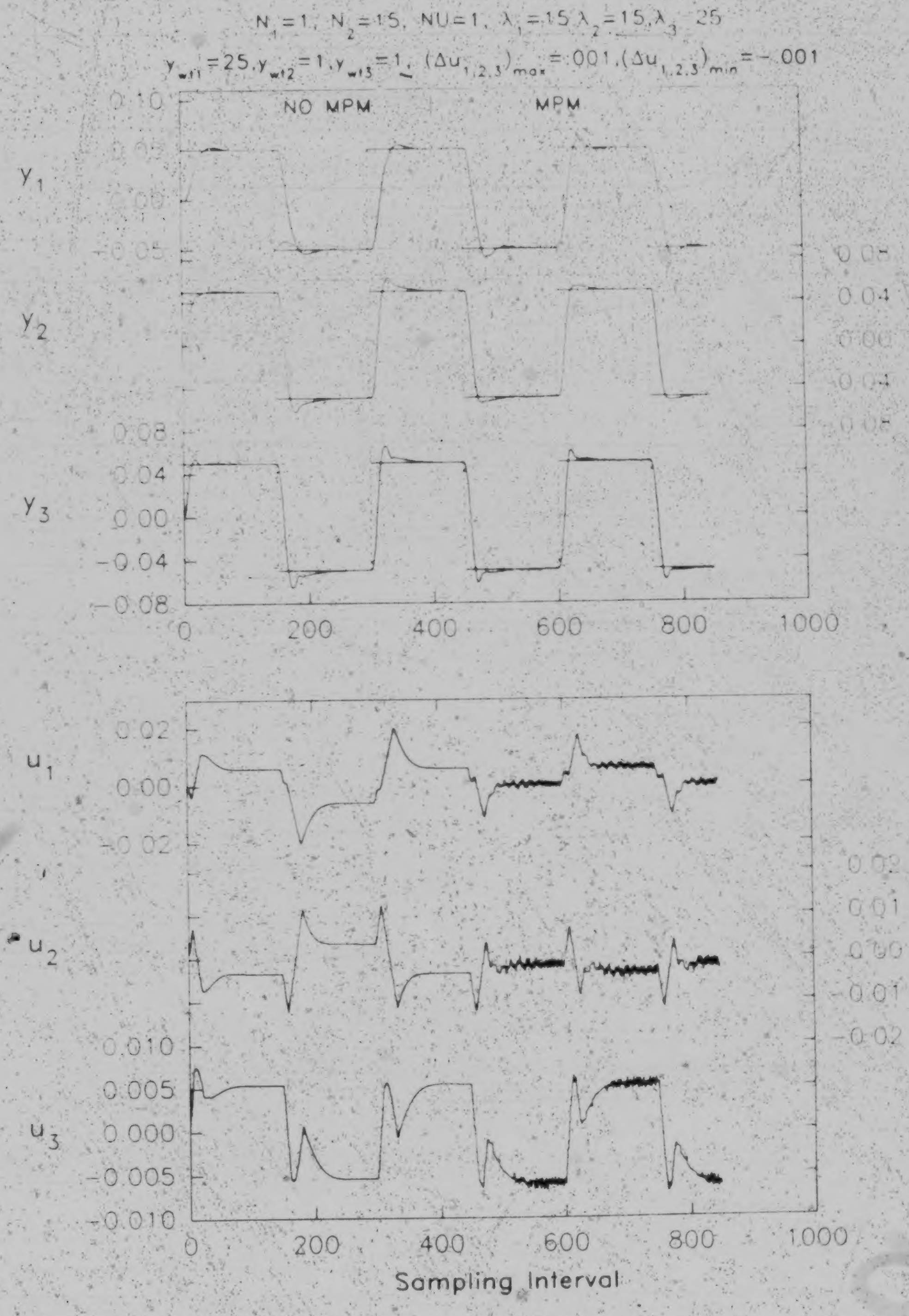


Figure 4.10: Control of Shell control problem with MPM in G_{11}

4.2.4.2 MPM in G_{11}, G_{22}, and G_{33}:

The model process mismatch was introduced in channels G_{11}, G_{22}, and G_{33} by changing the process steady state gains to 6.16, 5.15 and 8.53 from the actual values of 4.05, 5.72 and 7.20. Many simulations were performed for this case to study the effect of output weighting y_{wt}, Λ weighting and rate constraints. The 'tuning knobs' of MGPC were set to: $N_1 = 1$, $N_2 = 15$ and $NU = 1$ for all the simulations with MPM in G_{11}, G_{22}, and G_{33}. The model process mismatch for all the simulations was introduced at the sampling interval 400.

The first simulation was performed with 'default values' for the remaining tuning parameters of MGPC: $\lambda_1 = \lambda_2 = \lambda_3 = 0.01$, $y_{wt1} = y_{wt2} = y_{wt3} = 1$, $\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{max} = 0.001$ and $\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{min} = -0.001$. The time-trajectory plot for this simulation is shown in figure 4.11, which shows an offset in the tracking of outputs and therefore the performance of the controller with the above settings is not acceptable.

In the next simulation the input control weightings were changed to $\lambda_1 = \lambda_2 = 15$, $\lambda_3 = 25$ and the remaining settings were left unchanged. The time-trajectory plot for this tuning is shown in figure 4.12. The performance of this simulation is no better than the previous simulation (figure 4.11).

Output control weightings were changed in the next simulation. The setting of the controller were: $\lambda_1 = \lambda_2 = 15$, $\lambda_3 = 25$, and $y_{wt1} = 15$, $y_{wt2} = y_{wt3} = 2$, $\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{max} = 0.001$ and $\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{min} = -0.001$. The simulation of the Shell control problem with MPM in the diagonal elements of the transfer function matrix with this tuning is shown in figure 4.13. It is seen that the closed loop process is able to track the set-points without any offset and the performance is satisfactory. It must be noted that the degree of MPM in channel G_{11} is highest and also that the output weighting on this channel is highest.

To evaluate the contribution of input control weighting in achieving offset free stable

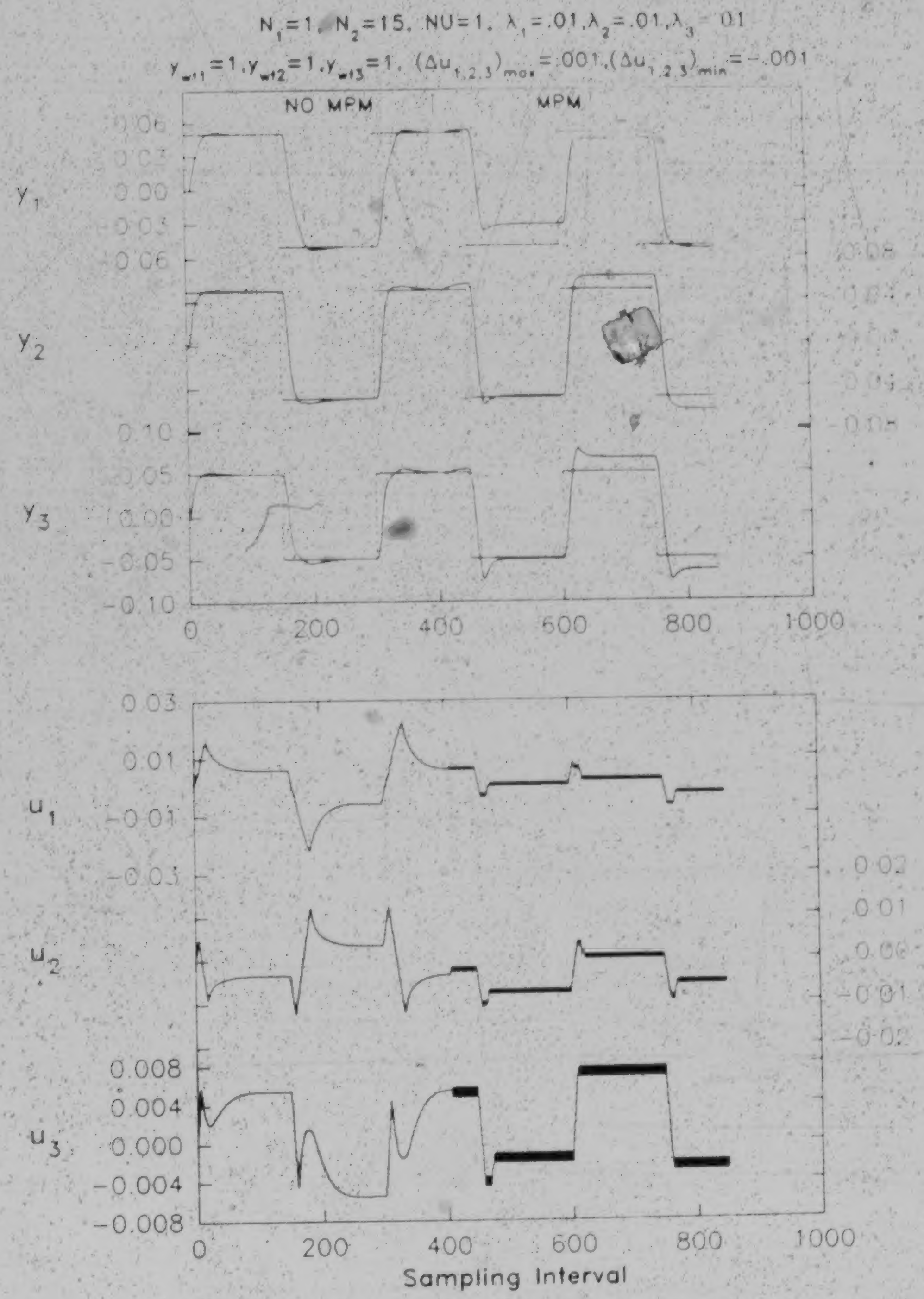


Figure 4.11: Control of Shell control problem with 'default' control setting for MPM in G_{11}, G_{22} and G_{33}

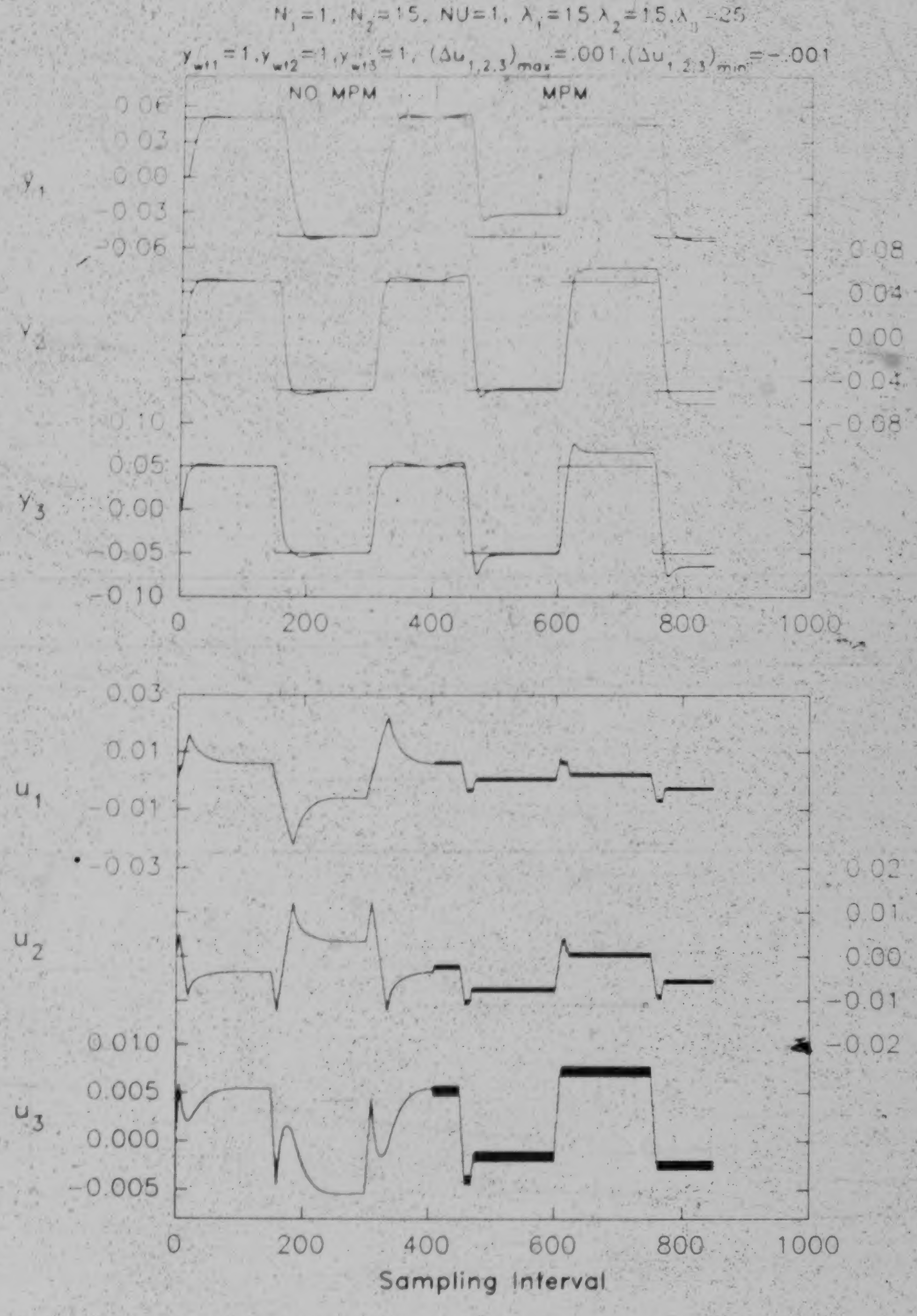


Figure 4.12: Control of Shell control problem with Λ weighting for MPM in G_{11}, G_{22} and G_{33}

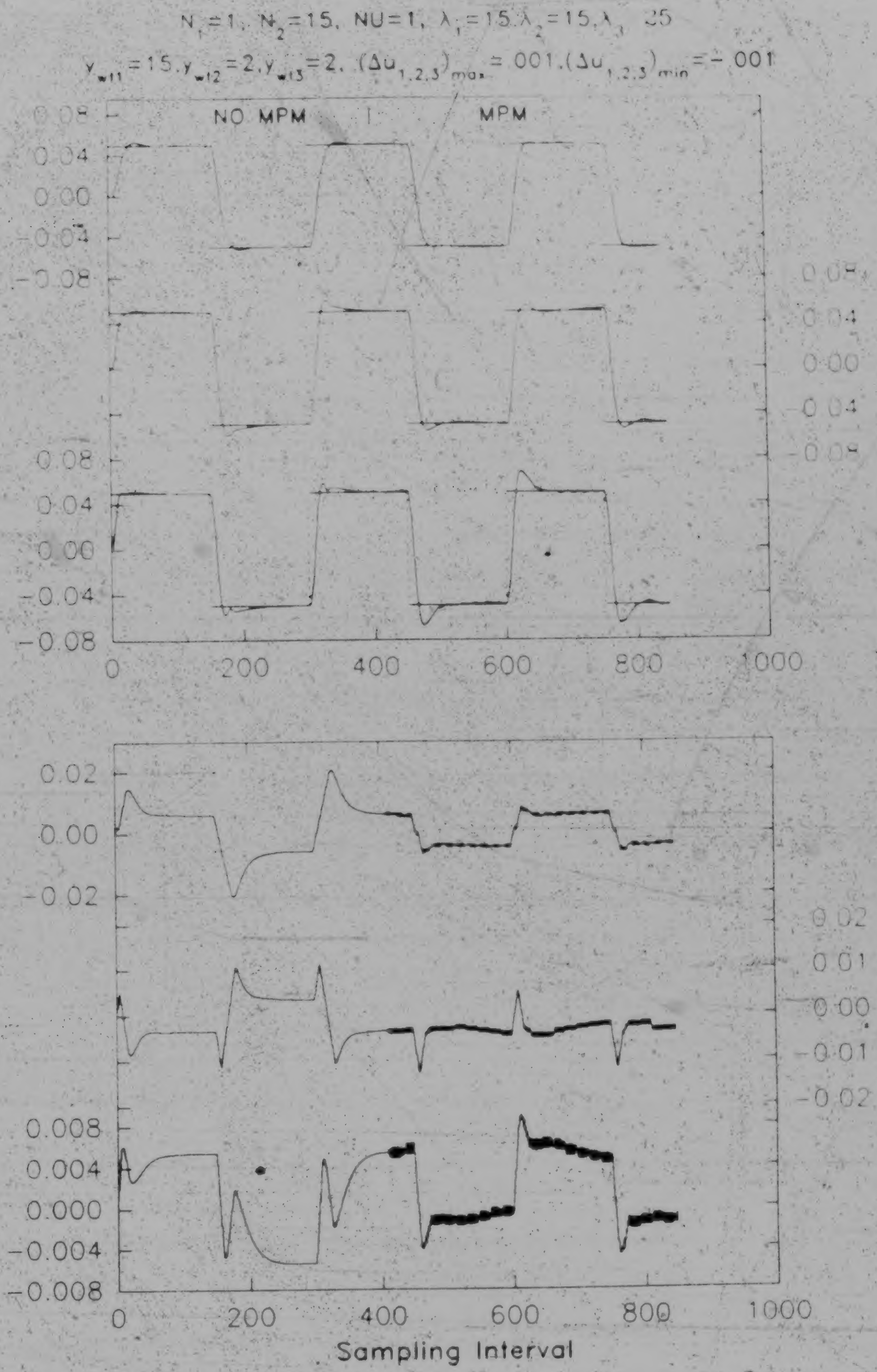


Figure 4.13: Control of Shell control problem with Λ and y_{wt} weighting for MPM in G_{11}, G_{22} and G_{33}

control, the control weightings were set to $\lambda_1 = \lambda_2 = \lambda_3 = 0.01$ and the remaining setting from the previous simulation were left unchanged. The MPM was introduced at sampling interval 400. The time-trajectory plot with these settings is shown in figure 4.14. The performance of this simulation is almost identical to the previous simulation results shown in 4.13. It can be inferred from the simulations in figures 4.12, 4.13 and 4.14 that Λ weighting, at least in this example, does not contribute to offset free control.

To evaluate the effect of rate constraints, a simulation was performed with the following settings of the controller: $\lambda_1 = \lambda_2 = 15$, $\lambda_3 = 25$, and $y_{wt1} = 15$, $y_{wt2} = y_{wt3} = 2$, $\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{max} = 0.01$ and $\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{min} = -0.01$. All the knobs are identical to the simulation in figure 4.13 except that maximum and minimum rate constraints were changed to +0.01 and -0.01 from +0.001 and -0.001. The MPM was introduced at sampling time 400. The simulation results for these settings are shown in figure 4.15. As is evident, the closed loop system becomes unstable after the MPM is introduced. Also note that the instability of the process results in the QP solving package, QPSOL, not converging to a feasible solution and thereby generating an error message. It can be conclusively said that the rate constraints can stabilize otherwise unstable systems by comparing the simulations in figures 4.13 and 4.15.

4.2.4.3 MPM in G_{11}, G_{21}, and G_{31}:

So far in this subsection, the MGPC algorithm could control processes with MPM by tuning the controller appropriately. This may not be possible for processes with large MPM in different channels. This point is illustrated by a simulation of the process with MPM in channels G_{11}, G_{21}, and G_{31} by changing their steady state gains of the process to 6.16, 2.10 and 7.49 from the actual values of 4.05, 5.39 and 7.49. All the three channels have MPM of at least 50% of the actual gains. After a lot of trial and error, the following values were selected for tuning the controller: $\lambda_1 = \lambda_2 = \lambda_3 = 3$, and $y_{wt1} = 2.5$, $y_{wt2} = 4.0$, $y_{wt3} = 3.5$,

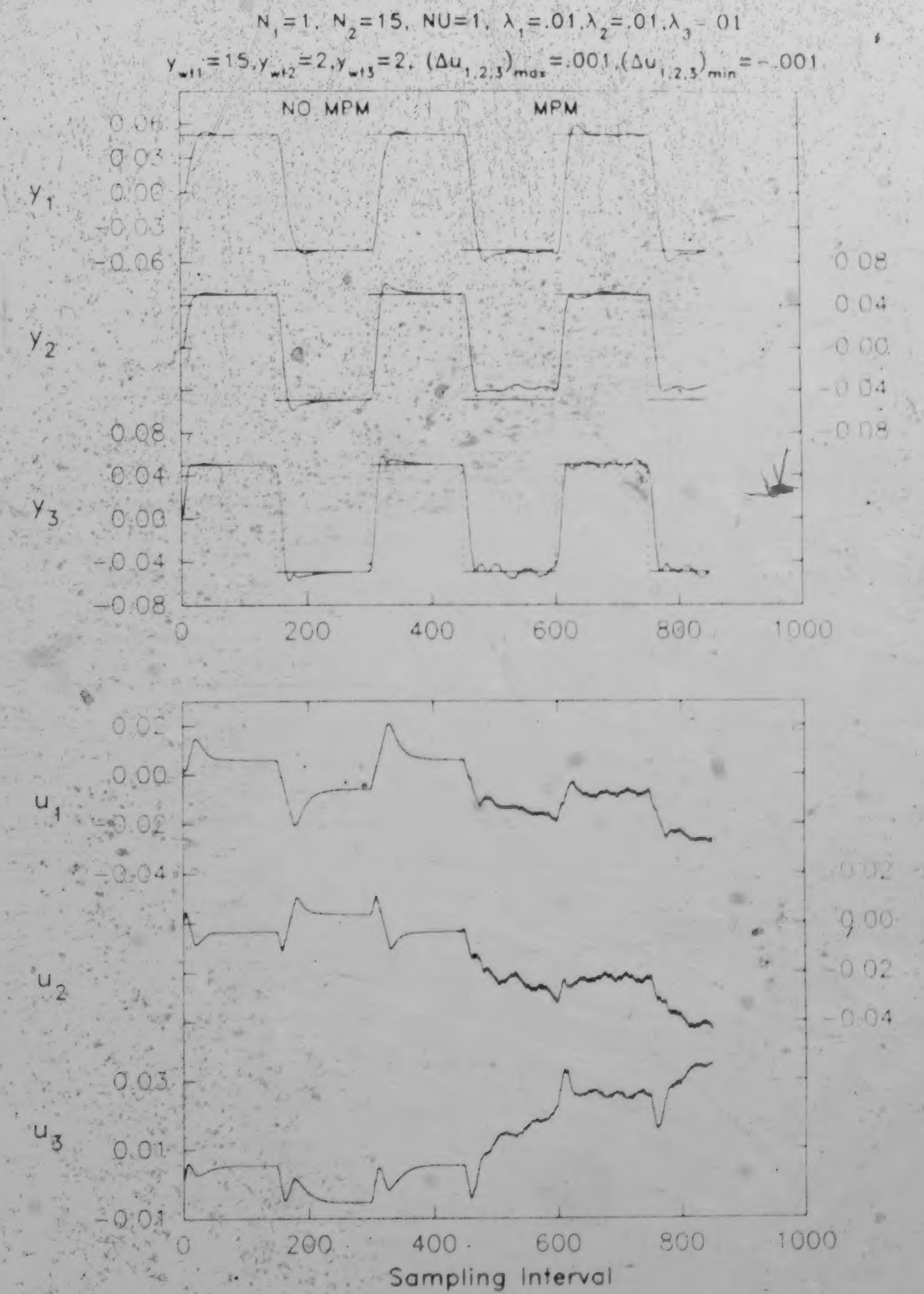


Figure 4.14: Control of Shell control problem with y_{wt} weighting for MPM in G_{11}, G_{22} and G_{33}

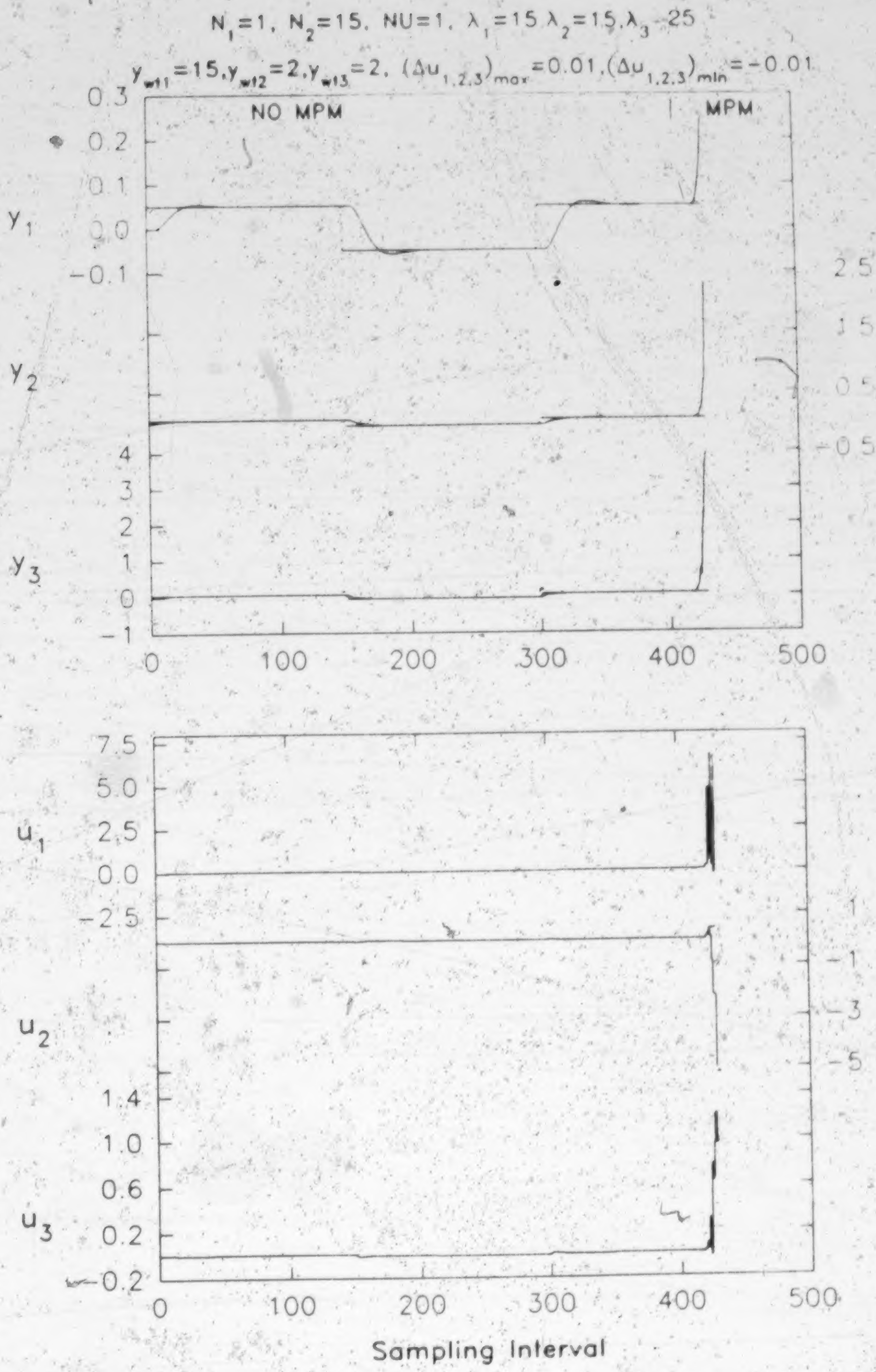


Figure 4.15: Control of Shell control problem with relaxed rate constraints for MPM in G_{11}, G_{22} and G_{33}

$\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{max} = 0.0005$ and $\{\Delta u_1 = \Delta u_2 = \Delta u_3\}_{min} = -0.0005$. The MPM was introduced at sampling interval 600.

The time-trajectory plot for this simulation is shown in 4.16. It is observed that the controller is integrating the error in the outputs in a 'wrong' direction. This is not surprising because the controller is trying to integrate the error between the setpoints and the process outputs using a model with gain model/process mismatch of more than 50% in the elements G_{11}, G_{21}, and G_{31} of the process transfer function matrix.

So far all the simulation performed for evaluating the effects of MPM were with the $T(q^{-1})$ polynomial equal to identity. $T(q^{-1})$ is a low pass filter and cannot contribute in achieving offset free control. This is verified by repeating the previous simulation (figure 4.16) with $T(q^{-1})$ equal to $1 - 1.6q^{-1} + 0.64q^{-2}$. The time-trajectory plot for this simulation is shown in figure 4.17. The plot shows that the offset between the outputs and setpoints, however slowly, is still increasing with sampling intervals.

4.2.4.4 Discussion

The discussion here is based on the simulation results in controlling the oil fractionator of the Shell control problem. Though the results are example specific, they may also be applicable to other systems.

Effect of output (y_{wt}) weighting:

This is an important tuning knob for controlling MIMO systems. From the simulations in figures 4.11, 4.12, 4.13 and 4.14, it is seen that the output weighting plays a major role in achieving offset free control for processes with MPM.

A general guideline for selecting the output weighting for system with MPM is: Larger output weighting terms for the outputs of channels with higher model process mismatch.

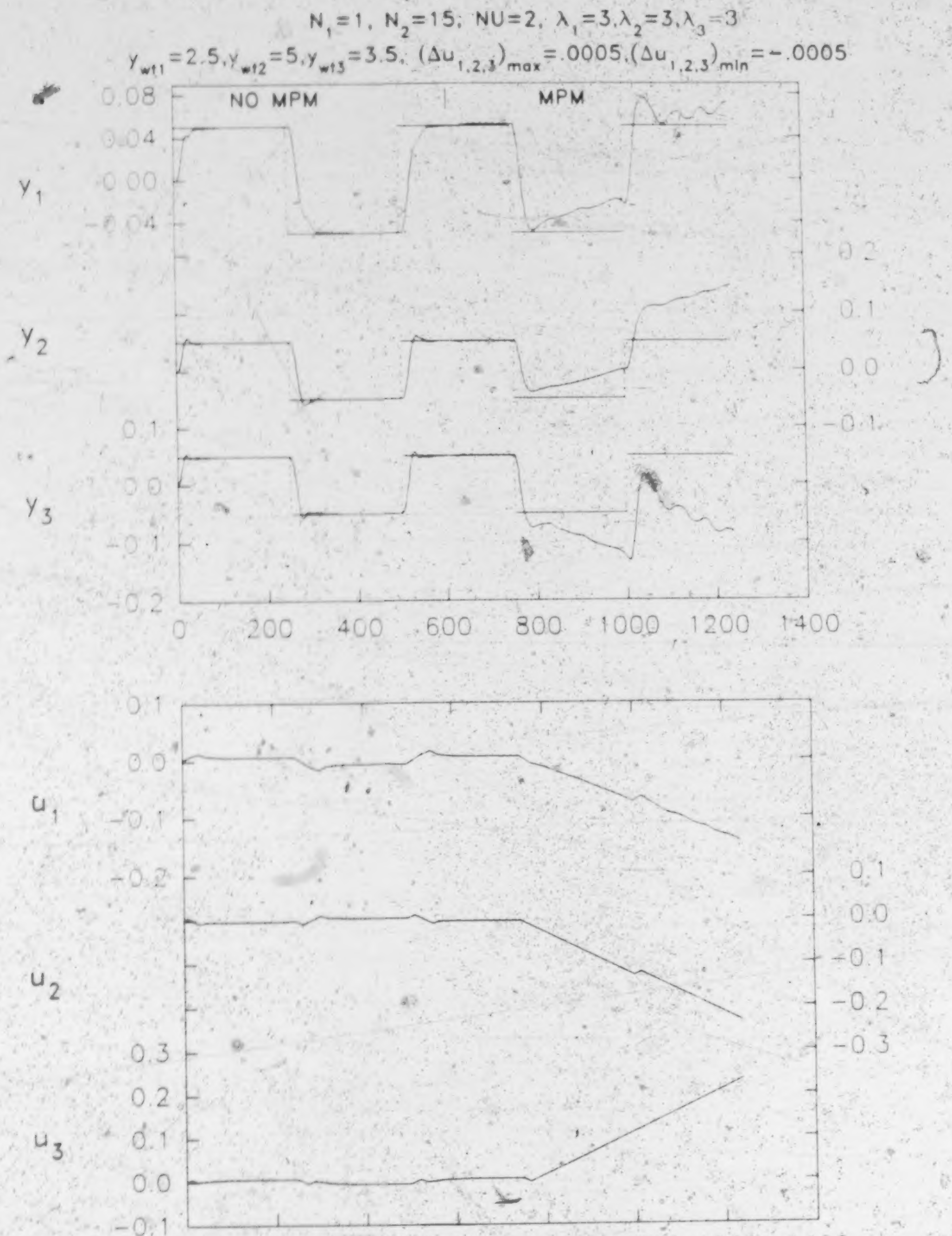


Figure 4.16: Control of Shell control problem rate constraints for MPM in G_{11}, G_{21} and G_{31}

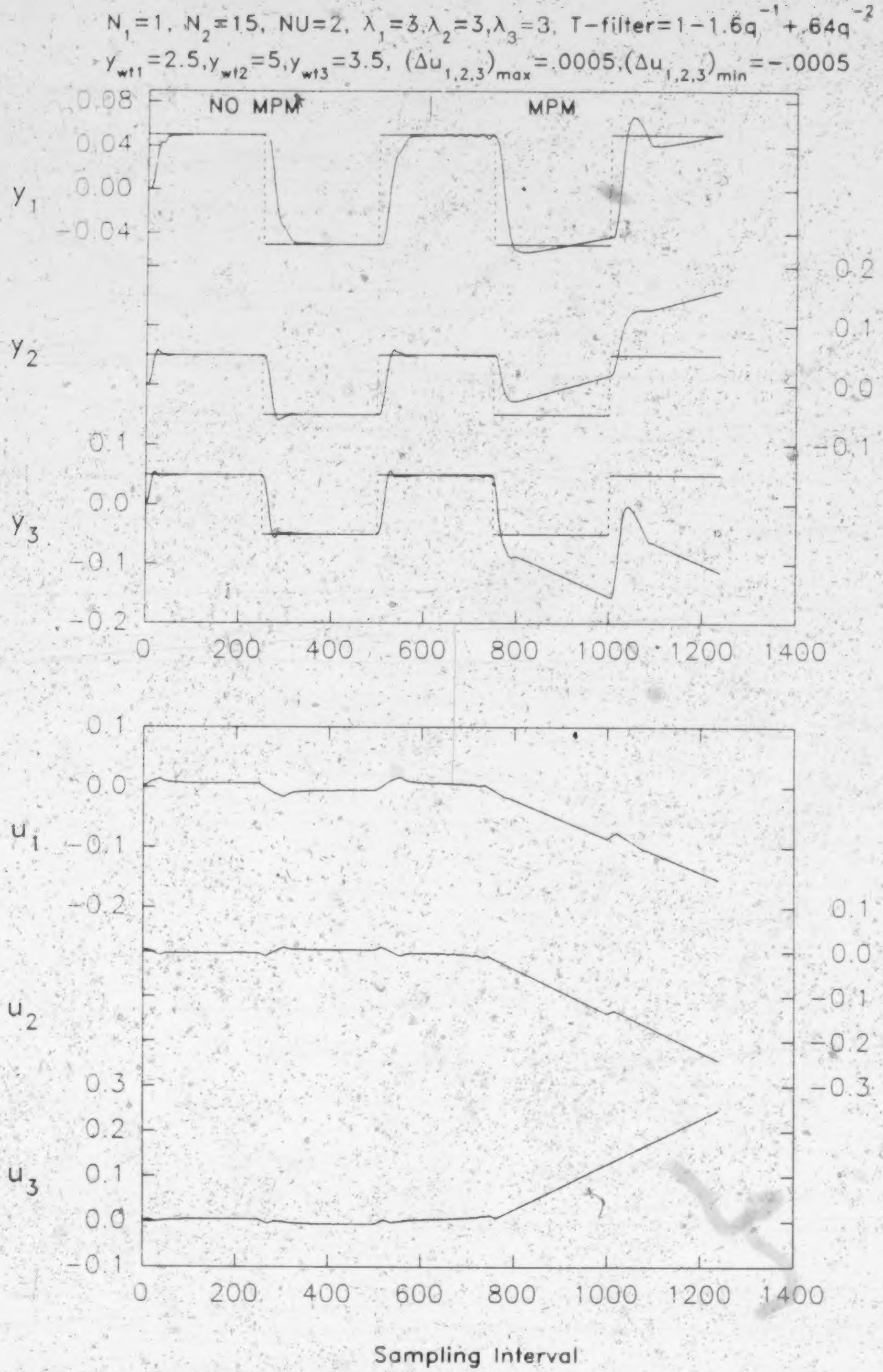


Figure 4.17: Control of Shell control problem with second order $T(q^{-1})$ for MPM in G_{11}, G_{21} and G_{31}

This guideline was used for tuning the simulation in figure 4.10. Also, the experimental runs described in chapter 5 support the above statement. Though there is no analytical proof to confirm this statement, it can be explained in a qualitative manner as follows: By giving higher output weighting to the channel with larger process MPM, the controller tries to minimize the error between the set-point and the output for that channel first. Then it tries to control the other channels which are easier for the control algorithm to control because of the relatively better models for these channels. At the same time, tracking the set-point for the channel with larger MPM can be achieved by the larger output weighting on this channel in addition to the integral action of the controller.

For arguments sake, consider the case of giving smaller output weighting to channels with relatively larger model process mismatch. In this case, the control algorithm will try to control the channels with lower MPM (and relatively large weighting) first and then control channel(s) with larger MPM. The controller will have to use its integral action to control the channel with larger MPM and this is not an easy task in a multivariable system. Unlike the SISO GPC, where integral action has to integrate the error in 'one direction' only, the MGPC controller has to integrate the error for different channels in 'different directions'.

In the presence of MPM, the integral action in a SISO system can eliminate the offset. However, for a MIMO system, the issues of MPM and offset elimination are fairly complex. The standard results of Shah *et al.* [46] for $NU = 1$ and $N_2 = \infty$ that results in a mean level integral control may not hold for the case of MIMO plant with MPM. The analysis of MIMO stability in the presence of MPM is a problem that is beyond the scope of this thesis.

Also it is mentioned in the literature [4] that the elements of the output scaling matrices should be such that: "Large weights correspond to small responses", *i.e.* scale the outputs such that the weighted error from each output is of equal order of magnitude. In the context of tuning y_{wt} guideline, it is equivalent to assuming that all the channels have

MPM proportional to the gain of the channel. If nothing is known about the degree of MPM in a process, then this method is appropriate for tuning the output weighting parameter.

The above logic also explains the performance of the MGPC algorithm in figure 4.16. The controller cannot handle large MPM (of $\geq$ 50% in the steady state gains) in all the output channels. The time-trajectory plots in figure 4.16 show that the controller is trying to integrate the error between the set point and the outputs—but in the 'wrong direction'. This may be due to the improper decoupling of the interactions between different outputs, due to the large amounts of MPM in different output channels.

This tuning knob can also be used for improving the decoupling properties of the closed loop system (Maurath *et al.* [34]).

Effect of Λ weighting and rate constraints:

From the simulations shown in the figures 4.14 and 4.15 it is clear that tighter rate constraints can stabilize some systems which could be otherwise unstable. The effect of weighting on input increments (Λ) gets eclipsed in the presence of tight rate constraints. However, from the simulations in figures 4.11 and 4.12, it can be seen that Λ weighting cannot improve the MGPC algorithms properties in handling MPM. The effect of Λ weighting in MIMO systems is very similar to SISO systems. Also, from the simulations in figures 4.13, 4.14 and 4.15 it is observed that rate constraints can handle unstable systems better than the Λ weighting. However, unlike rate constraints, *if* very large Λ weighting can stabilize the system, then it would result in a 'smoother' control action.

Limitations of fixed parameter MGPC

In the above simulations, the tuned MGPC could control the fractionator with reasonable MPM. However, for the simulation in figure 4.16, the controller could not achieve offset free control. This is because of large MPM (of $\geq$ 50% in the steady state gains) in all the output channels. The integral action of the controller was integrating the error in the 'wrong direction' and thus resulted in improper control.

For processes with such large model process mismatch (MPM), it is necessary to use adaptive MGPC algorithm. The fixed parameter MGPC cannot handle very large degree of MPM. Adaptive SISO GPC but not adaptive MGPC is considered in this thesis.

4.3 Conclusions

In this chapter, the performance of the MGPC algorithm was evaluated by simulating the oil fractionator of the Shell control problem [42]. The MGPC algorithm was implemented in programming languages 'C' and Fortran and the constrained cost function was solved by a "commercial type" quadratic programming optimization package, QPSOL [19]. The developed MGPC algorithm was implemented on the Shell control problem [42] and was evaluated for its performance in handling interaction between different outputs, load rejection properties–with and without feedforward compensation, and model process mismatch.

The closed loop performance of the simulated system shows fairly well-decoupled outputs. The interactions in the system could be further reduced by increasing the control horizon, NU. The control algorithm, without feedforward compensation, successfully rejected step load disturbances in the process. The performance of the MGPC algorithm with feedforward control can provide *dynamic* compensation for the disturbances and hence the performance is superior for load rejection. Many simulations were performed to evaluate the performance of MGPC controller in handling model process mismatch. The non-adaptive well tuned MGPC algorithm can handle reasonable amounts of MPM. The tuning parameter y_{wt} plays an important role in tuning the MGPC algorithm for different amounts of MPM in different channels of the process. The simulations also show that the non-adaptive MGPC cannot handle systems with large MPM in many channels of the process. Adaptive MGPC should be able to handle such processes. Adaptive MGPC has not been considered in this thesis.

Chapter 5

Experimental Application of GPC

The performance of constrained SISO and MIMO GPC on simulated processes was the subject of the previous chapter. Whereas evaluation by simulation is helpful in giving an insight on the performance of the algorithms under different parameter settings, there is no substitute for evaluation of the algorithm under real conditions–*i.e.* experimental evaluation on a real process. Such runs help to evaluate the performance of an algorithm for unmeasurable dynamics, for non-linearities and model-process mismatch.

The adaptive GPC with and without constraints was implemented on SISO systems and the fixed parameter GPC was implemented on a two-input two-output system. For comparison with MGPC, an experimental run was also conducted with two multiloop PID controllers on the two-input two-output system.

This chapter outlines the hardware used for implementing the control algorithm in section 5.1; sections 5.2 and 5.3 describe the process and the experimental evaluations conducted on the stirred tank heater for SISO and MIMO configurations respectively; section 5.4 describes the physical setup and the experiments performed to control the mean arterial blood pressure of a dog with a constrained adaptive SISO GPC.

5.1 Implementation Details

Implementation of a long range predictive control algorithms requires more sophisticated hardware and software than conventional process control algorithms. It is necessary to have a real-time computer system with multi-tasking feature to implement an advanced control scheme. In this section, the hardware and type of software used for implementing the control algorithms will be described.

The control algorithms were implemented under the QNX operating system on an IBM PS/2 model 70 personal computer (PC). QNX is a real-time, multi-tasking, multi-user operating system. Further details of the operating system QNX (Version 2.12) can be obtained from the manuals provided by Quantum Software Systems Ltd [61]. This operating system loaded onto a portable PC facilitates easy transportation of the hardware and control algorithm to the application site. Lau [28] fully describes the justification for selecting this operating system for implementing the control algorithms.

The control algorithms were implemented on the equipment with the help of a software package MULTICON (for **MULTI**-purpose **CON**trol) developed by Qiu [43]). MULTICON, developed at the University of Alberta, Canada, is a real time executive program that runs under QNX operating system. It is created to support the following:

1. Execution of user tasks and to control their activity for real time control application.
2. Creation of a channel for process operator communication.
3. Provide a pool as a common data area for all user tasks and a set of utility programs to access the data.
4. Display on-line graphics task.
5. Perform tasks scheduling.
6. Handle I/O device drivers.

With all the above features, MULTICON lends itself as a flexible medium for a control engineer to develop, test, and compare different control algorithms without the knowledge of the message system of QNX. For further description of the program, the reader is referred to Lau [28] and Qiu [43]

The computer communicates with the process through the Opto-22 interface. The Opto-22 behaves as a front end computer to offload the host computer (IBM PS/2 model 70) of the I/O tasks. It is connected to the host machine through a RS-232 serial line. Its responsibility includes:

1. optically isolating the process from the computer,
2. continuously scanning the I/O channels and providing signal conversions for inputs and outputs (*eg.* analog to digital conversion), and
3. responding to requests from the host computer.

Whenever the host computer needs to access a process variable, it sends a command to Opto-22. For example if the computer wants to sample the output, Opto-22 sends the most recent digital data from the appropriate transducer and loads it onto the serial line to send the data to the host computer. A complete set of communication protocols to operate Opto22 is provided in Optomux Analog User's Manual [62].

All the experiments described in this chapter were conducted with the above described operating system QNX, executive program MULTICON, and the process interface Opto-22. The software used for the simulations in chapter 4 was also used for the experimental runs. As described in the earlier chapter, mixed language programming was used to write the control algorithms. Mixed language programming was necessary due to the use of optimization package QPSOL [19] from Stanford University, which was in FORTRAN. The control algorithms were mainly programmed in C, to facilitate the use of dynamic memory allocation. The control algorithm is described in section 4.1 and the MATLAB version of the algorithm is in appendix C.

5.2 Control of Stirred Tank Heater with Adaptive SISO GPC

Experiments were conducted on a continuously stirred tank heater (CSTH) to evaluate the performance of GPC. This section describes the experiments conducted for evaluating the adaptive SISO GPC. Section 5.2.1 describes the process and its configuration for performing the SISO experimental runs and section 5.2.2 describes the results of the experiments that were conducted.

5.2.1 Process Equipment

A stirred tank heater/mixer was used for conducting the SISO and MIMO experimental runs. The SISO configuration of the equipment is illustrated schematically in figure 5.1.

As shown in figure 5.1, cold water flows into the glass tank from the cold water inlet pipe. The water in the glass tank is heated by a steam coil and flows out through a long copper tube, which has thermocouples at varying distances from the tank. The steam flow rate through the heating coil and the cold water flow rate into the tank can be controlled by two separate pneumatic valves. In this setup, the temperature of the water flowing out of the glass tank was controlled by steam valve opening, expressed as a percentage, as the manipulated variable. The cold water flow rate was used as a disturbance variable. The level of water in the tank was maintained constant by manipulating the exit water flow rate. Standard or nominal operating conditions for the stirred tank heater are listed in table 5.1.

Inlet Water Temperature	5	°C
Inlet Water Flow rate	53	cm^3/s
Water Level	24	cm
Steam Valve Opening	50%	
(Resulting) Outlet Water Temperature	35	°C

Table 5.1: Nominal operating conditions for SISO configuration of stirred tank heater

While the tank is equipped with a mixer, the water is still not "well-mixed" as the inlet water flows into the tank on the same side as the exit tubing connection. However,

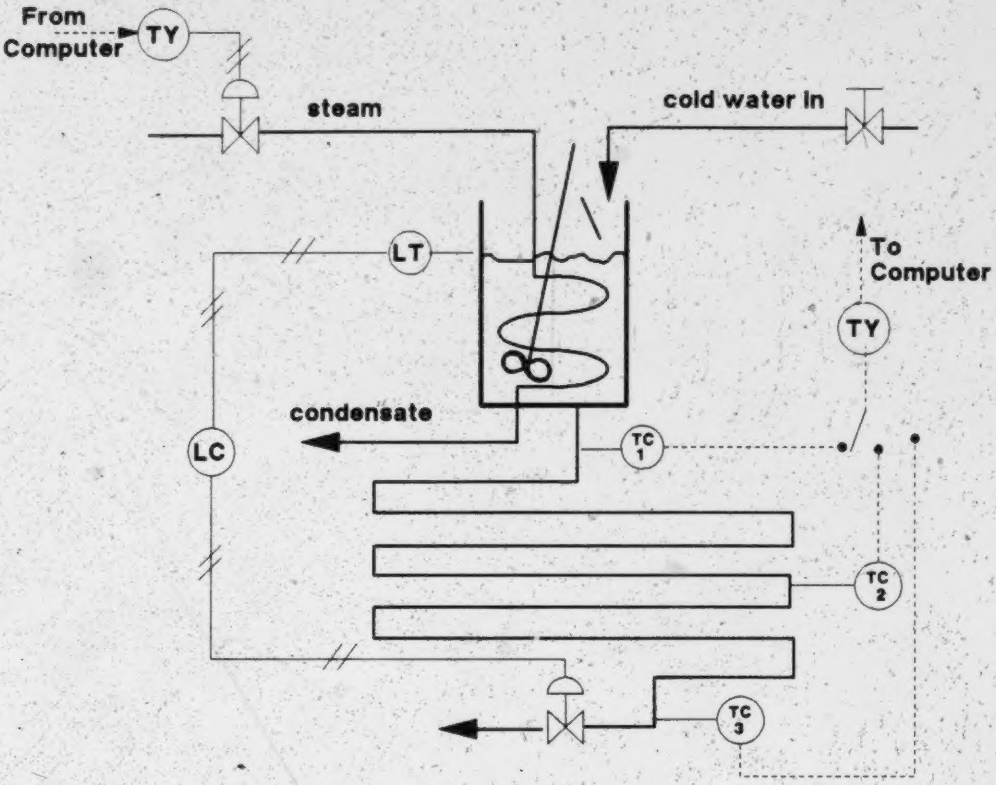

From Computer
TY
steam
cold water in
LT
To Computer
TY
condensate
LC
TC 1
TC 2
TC 3

Figure 5.1: Schematic diagram of Continuously Stirred Tank Heater for SISO configuration

when a deflector is put in place to direct the water to the wall of the tank and away from the outlet tubing connection the resulting water temperature is free of noise and can be considered as well mixed.

SISO runs were implemented with an adaptive GPC where the model of the process was identified on-line. The physical process has dominant first order dynamics with a time delay, which depends on the thermocouple chosen on the long copper tube. The sampling interval was chosen as 8 seconds for all the SISO experimental runs. This sampling interval is approximately 1/6th of the dominant process time constant and 1/5th of time delay when using thermocouple # 2, on the copper tube of the stirred tank heater.

5.2.2 SISO Experimental Runs

This section describe the various experimental runs conducted to test the adaptive SISO GPC and fixed parameter GPC. The experimental runs were conducted with two basic objectives in mind. First, to demonstrate the performance of the control algorithm. Secondly, to compare the performance of a constrained and an unconstrained controller.

As explained earlier, the SISO experimental runs were implemented with the temperature as the controlled variable, the steam valve opening as the manipulated variable and the change in cold water flow rate as the disturbance variable. The first thermocouple on the copper tube of the equipment (figure 5.1) was used for all the SISO experimental runs with a sampling time of 8 seconds.

The long range adaptive identification (LRPI) algorithm of Shook *et al.* [47,48,49] in conjunction with a recursive least squares algorithm and the directional forgetting factor of Kulhavy [26] was used to estimate the model of the plant, for all the SISO runs. The LRPI algorithm is briefly described in section 2.5. The estimation software is capable of changing the order of model being estimated on-line. The LRPI on line model estimator was used for identifying the A polynomial with 1 parameter and B polynomial with 2 parameters (equation 2.1) *i.e.* identifying a first order process with a possible non-integer sample time-delay. During the first 25 sampling instances the control signal was manually switched between user-specified values (every 4 or 5 sampling instances). This was sufficient for the parameter estimator to obtain a reasonable model.

The control algorithm is also capable of handling the change in order of model on line. When the controller is switched on, bumpless transfer is automatically achieved, due to the incremental nature of the control law. The control algorithm can handle constraints on amplitude and rate of change of manipulated variables as well as constraints on output. The constraints on amplitude of the manipulated variable are used for protecting the controller from "integral windup". For all the SISO experimental runs, the minimum and maximum amplitude constraints were set to 0% and 100% valve opening–the physical lower and upper bounds of the manipulated variable.

Section 5.2.2.1 illustrates the effect of change in rate constraint on the manipulated variable and section 5.2.2.2 compares the performance of constrained and unconstrained GPC algorithms. The tuning parameters of GPC for all the SISO experiments were set to $N_1 = 1$, $N_2 = 10$, $NU = 1$, $\lambda = 0.005$, default values as suggested in Mohtadi [35]. A T-filter of $1 - 0.8q^{-1}$ was used by the control algorithm, as suggested by McIntosh [30]. The model estimator, LRPI algorithm, uses a time varying filter.

5.2.2.1 Effect of Rate Constraint

To illustrate the effect of a rate constraint on the performance of the controller an experimental run was started without a rate constraint. After allowing 25 sampling intervals for model estimation, the GPC controller was turned on. After 170 sampling instances, rate constraints of $-10\% \leq \Delta \mathbf{u} \leq +10\%$ were imposed as upper and lower bounds on the change of manipulated variable *i.e.* the valve could open or close by a maximum of 10% per sampling interval. The constraints were further narrowed down to $-5\% \leq \Delta \mathbf{u} \leq +5\%$ at sampling interval 265 and then to $-2\% \leq \Delta \mathbf{u} \leq +2\%$ at sampling interval 325. The performance of this run is shown in figure 5.2. The results in this figure demonstrate the ability of the estimation and control algorithm to perform their respective functions. From the experimental run it is seen that the performance of $-10\% \leq \Delta\mathbf{u} \leq +10\%$ rate constrained GPC is almost as dynamic as the unconstrained GPC algorithm. When the rate constraints were narrowed down to $-5\% \leq \Delta\mathbf{u} \leq +5\%$ and subsequently to $-2\% \leq \Delta\mathbf{u} \leq +2\%$, the response of the system becomes relatively sluggish *i.e.* the time constant of the closed loop system increases as the constraint is narrowed down.

5.2.2.2 Performance Comparison of Rate Constrained and Unconstrained GPC

To compare the performance of the constrained and unconstrained GPC, a process with noisy output signal was created by removing the inlet water deflector and stopping the stirring of water in the tank. Two separate runs were performed with constrained and unconstrained GPC algorithms.

Performance of the unconstrained GPC is shown in figure 5.3 and of the constrained GPC is shown in figure 5.4. A positive step disturbance in the cold water inlet flow rate was implemented at sampling interval 151 followed by a negative step disturbance at sampling interval 176, for both the runs. It is clear from the responses of unconstrained and constrained GPC (figure 5.3 and 5.4) that the load rejection characteristics of the constrained case is better. Also, the control action implemented by the constrained GPC is not as large as the unconstrained GPC. Although it is not possible to isolate the exact role of the constraints in the experimental runs, it can be inferred that the aggressive control action is restricted by the constraints and is the major reason for the better performance.

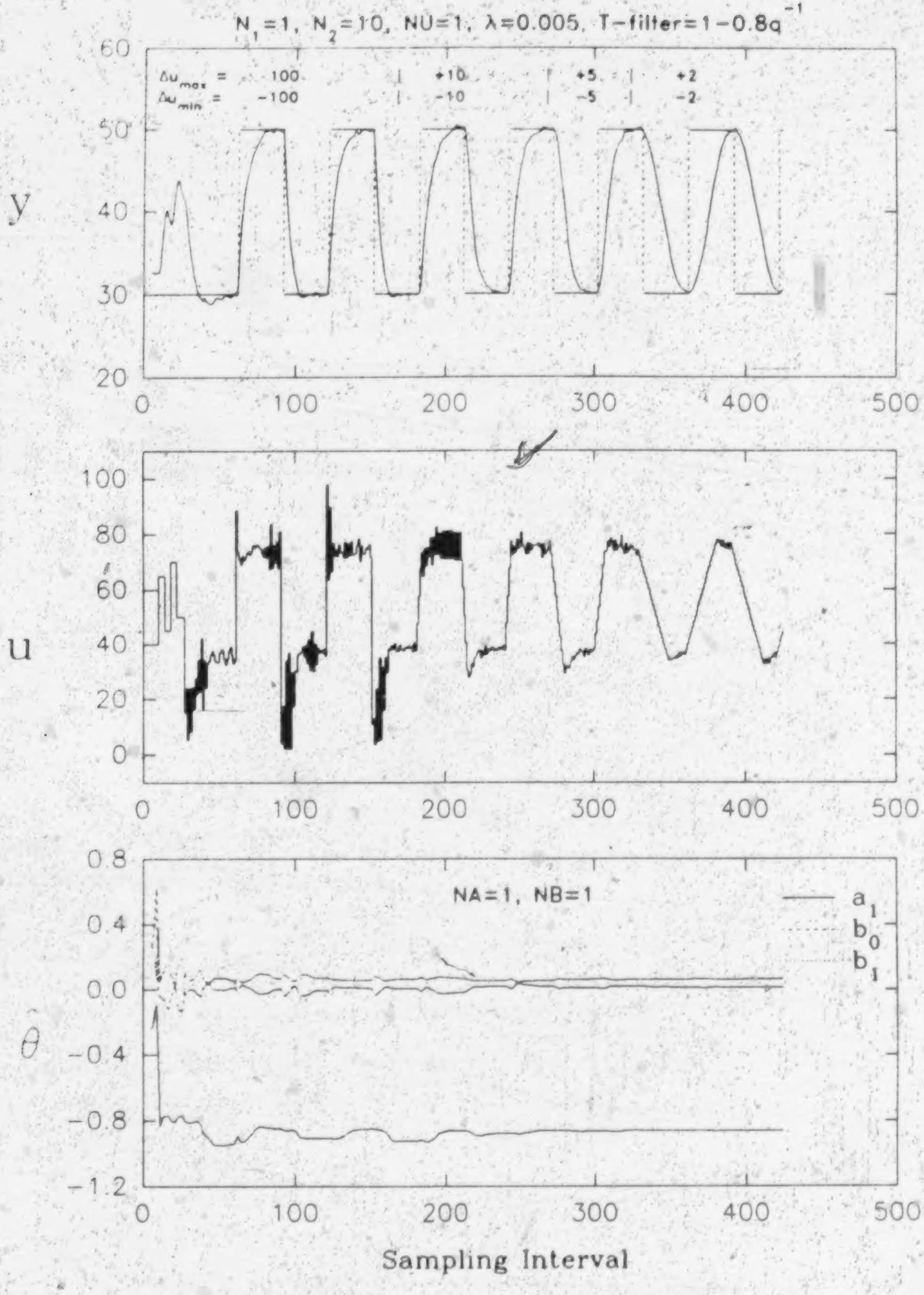


Figure 5.2: The effect of rate constraints on the control of the stirred tank heater in SISO configuration

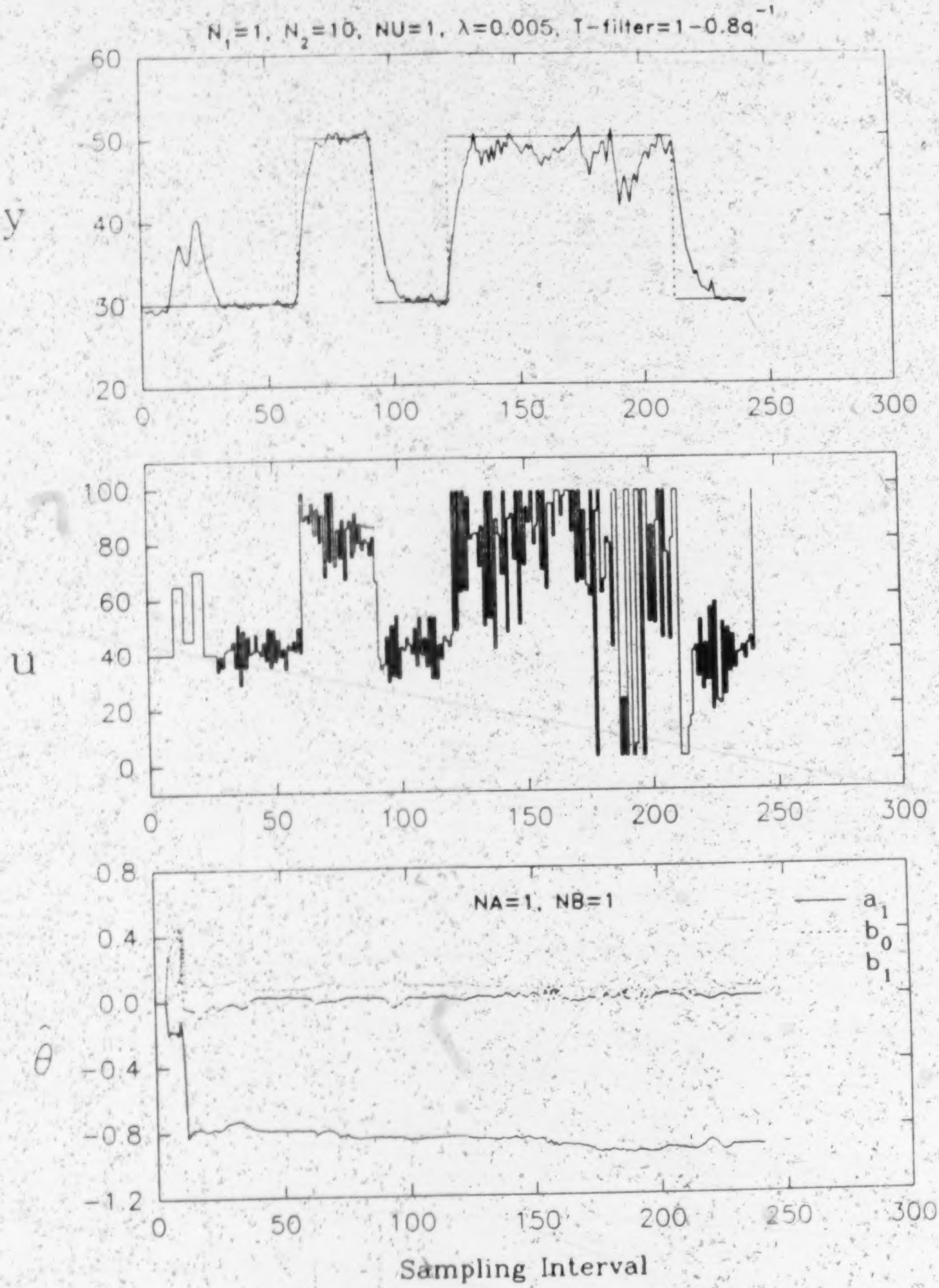


Figure 5.3: Performance of an unconstrained SISO GPC on the stirred tank heater

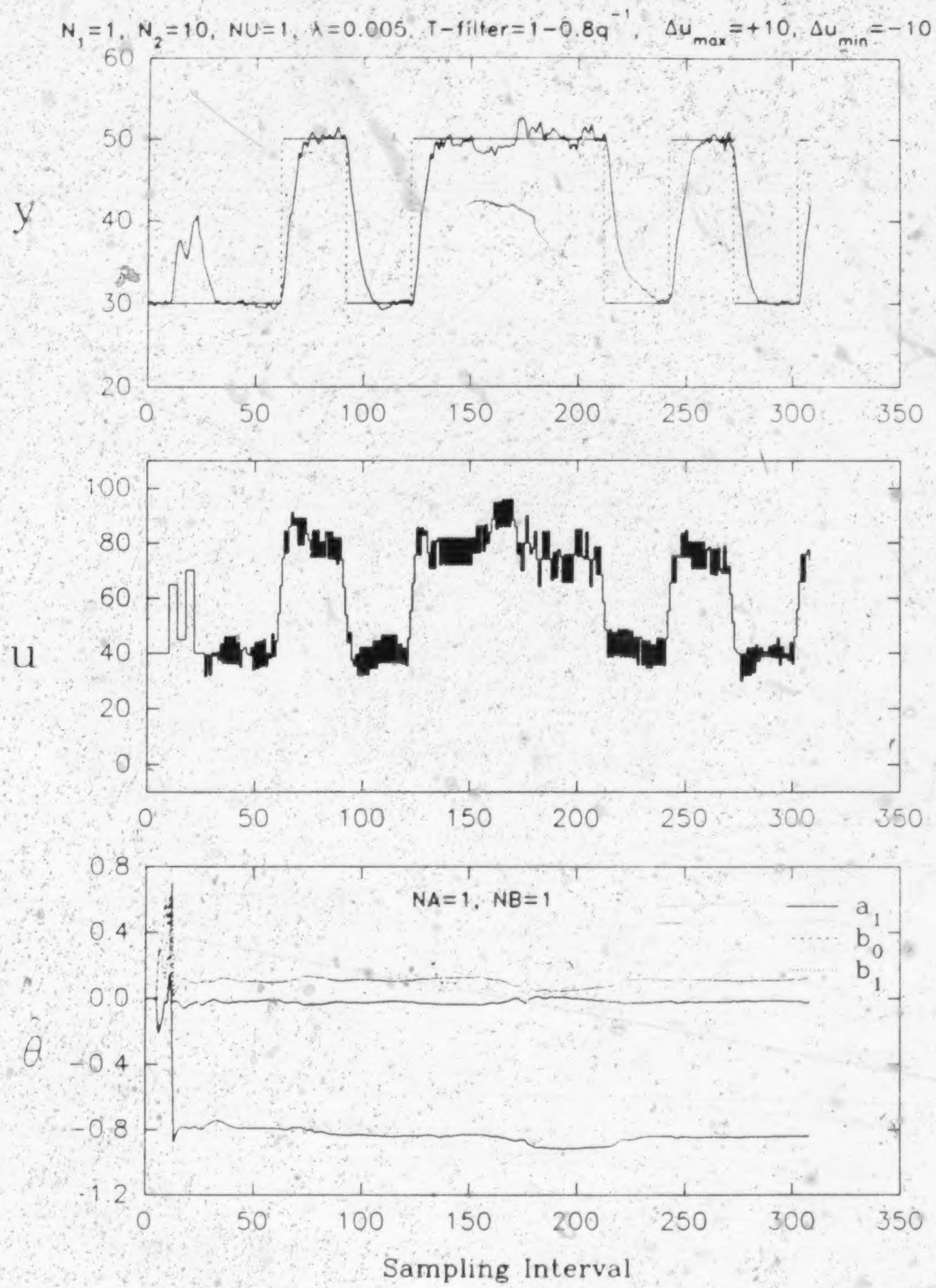


Figure 5.4: Performance of a rate constrained SISO GPC on the stirred tank heater

5.2.2.3 Effect of Output Constraints

An experimental run was conducted on the stirred tank heater to evaluate the effects of output constraints. The control algorithm was implemented to regulate the temperature of the process at 43 °C in the presence of step load disturbances created by manually changing the inlet flow rate of cold water. The steady state gain and the dynamics of the process change with inlet flow rate of the cold water. The LRPI algorithm was used for estimating the model of the process. To isolate the effects of output constraint in the control algorithm from the plant model update, the estimation of the process was stopped at sampling interval 43 *i.e.* once a reasonable model of the process was identified, the estimator was switched off. Thus, evaluation of the output constraints was performed with non-adaptive SISO GPC with the tuning parameters set to $N_1 = 1$, $N_2 = 10$, $NU = 1$, and $\lambda = 0.05$.

The GPC controller was switched on at sampling interval 35. Step load disturbances were introduced by varying the inlet cold water flow rate (**d**). The time trajectory plot for this run is shown in figure 5.5. While the controller was unconstrained, positive and negative step changes were given in the disturbance variable, **d**, at sampling intervals 50, 64 and 81. The experimental run was conducted with the controller regulating the temperature at 43 °C. Constraint on maximum process output (y_{max}) was set to 43.5 °C at sampling interval 95. The load rejection characteristics of the system for a positive step change in the disturbance at sampling interval 100 is similar to the performance of unconstrained system, as the output does not hit the constraint. When a negative step disturbance was introduced in the process at sampling interval 120, the process output hits the y_{max} constraint and the control action reacts vigorously. Load rejection of the system when the process output hits the y_{max} constraint is more dynamic than the unconstrained system. This exercise was repeated by changing the minimum output constraint (y_{min}) to 42.5 °C and by changing the maximum output constraint (y_{max}) to 54 °C (*i.e.* unconstrained y_{max}). In this case also the load rejection of the system when process hits the y_{min} constraint is more dynamic.

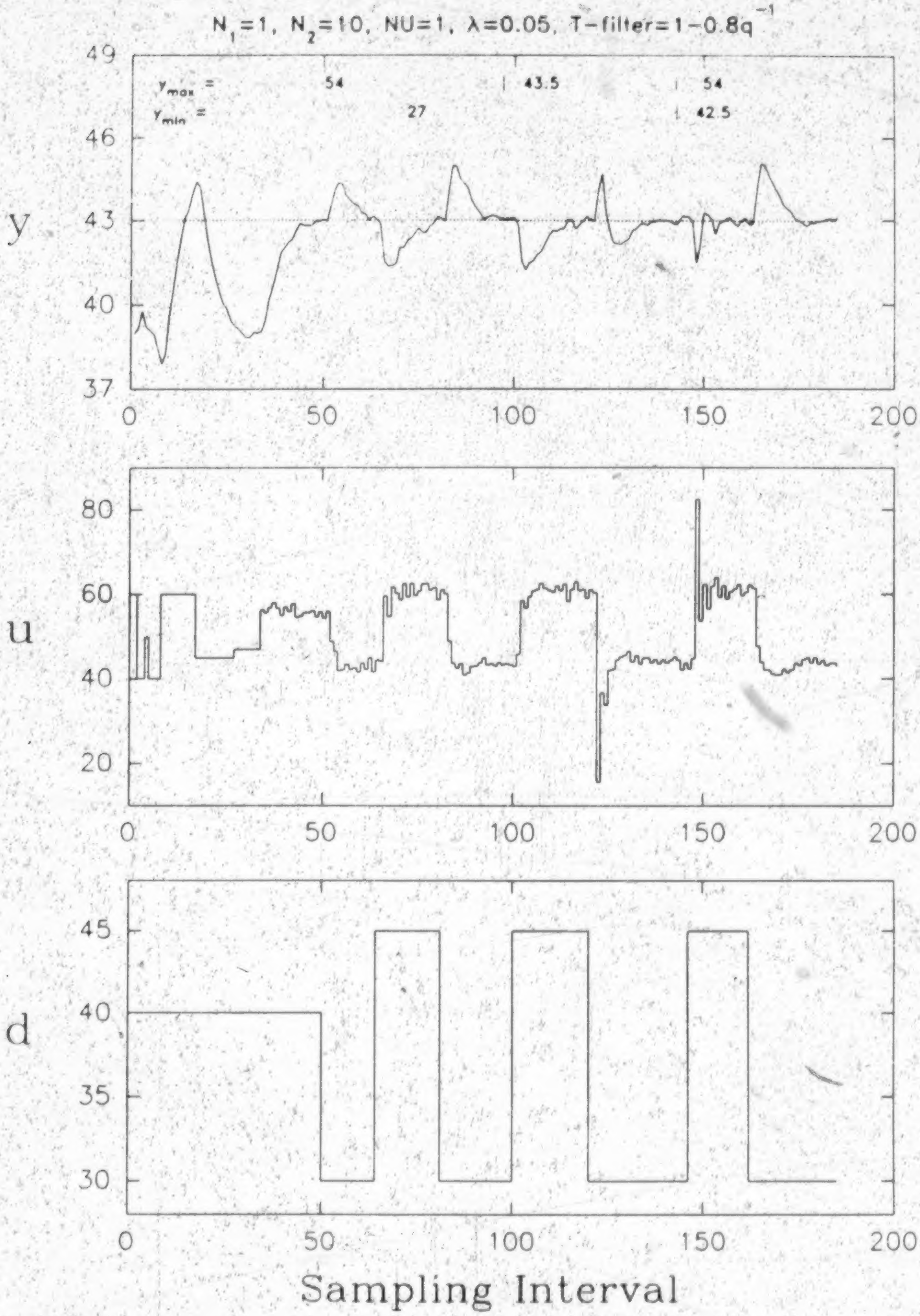


Figure 5.5: Performance of a output constrained SISO GPC on the stirred tank heater

This run illustrates that whenever the process output hits the output constraint, than the control algorithm will react vigourously to bring back the plant output in feasible region of the constraints.

5.3 Control of Stirred Tank with Fixed Parameter MGPC

Experimental runs for evaluation of MGPC were conducted on a two-input two-output system. The objectives of the experimental runs are to evaluate the performance of MGPC algorithm for different tuning parameters and to compare the performance of constrained and unconstrained MGPC. The performance of MGPC algorithm is also compared to the performance achieved by a multiloop PID controller.

A fixed parameter MGPC was used for the experimental runs described in this section. However, if a multivariable model identifier is implemented at a future date then provision for implementing adaptive MGPC exists in the implemented MGPC package.

Section 5.3.1 describes the stirred tank for the MIMO experimental runs and section 5.3.3 describes the various experiments conducted for evaluating the control algorithms.

5.3.1 Process Equipment

The stirred tank used for SISO experimental runs was configured differently for implementing MIMO experimental runs. Figure 5.6 shows a schematic representation of the process. The equipment shows two inlet pipes leading into the glass tank. Cold water flows through one pipe and hot water flows through the other. The supply of the cold and hot water is controlled by the Utility Services of the University of Alberta, and the temperatures are maintained between 5 to 10 degrees centigrade and 45 and 50 degrees centigrade respectively. The water from the glass tank flows out through a long copper tube, which has thermocouples at varying distances from the tank. The exit flow rate of water from the copper tube is gravity based and not controlled. The nominal operating condition for the stirred tank mixer are summarized in table 5.2.

Inlet cold water temperature	17	°C
Inlet hot water temperature	50	°C
Cold water valve opening	20.8%	
Hot water valve opening	10.9%	
Inlet cold water flow rate	49	cm^3/s
Inlet hot water flow rate	58	cm^3/s
Water level in the tank	60%	
(Resulting) Outlet Water Temperature	40	°C

Table 5.2: Nominal operating conditions for MIMO configuration of stirred tank heater

The aim of the experiments was to control the temperature (y_1) and level of water (y_2) in the tank by manipulating the percentage opening of cold (u_1) and hot water (u_2) valves. The level of water in tank was specified as percentage of maximum height of the tank. The open loop step response of this two-input two-output system is shown in figure 5.7 for the y_1—u_1 system, in figure 5.8 for the y_1—u_2 system, in figure 5.9 for the y_2—u_1 system, and in figure 5.10 for the y_2—u_2 system.

A more practical alternative for the simultaneous temperature and level control of the tank would be to have three manipulative variables, namely the inlet hot water flow rate, the inlet cold water flow rate and the outlet flow rate of water from the stirred tank mixer. This 2×3 problem can also be handled by the existing GPC software. However, for ease of application (in terms of minimum changes to control and instrumentation hardware) the 2×2 problem described earlier was considered. This 2×2 problem, in itself proved sufficient challenge for the controller, *i.e.* it was not a trivial problem.

The following transfer function matrix was used as the model of the process with temperature measured at thermocouple # 1:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \frac{-0.35e^{-4s}}{40s+1} & \frac{0.48e^{-4s}}{44s+1} \\ \frac{.031}{s} & \frac{.0223}{s} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{5.1}$$

where

- y_1 is temperature of the resultant water in °Centigrade,

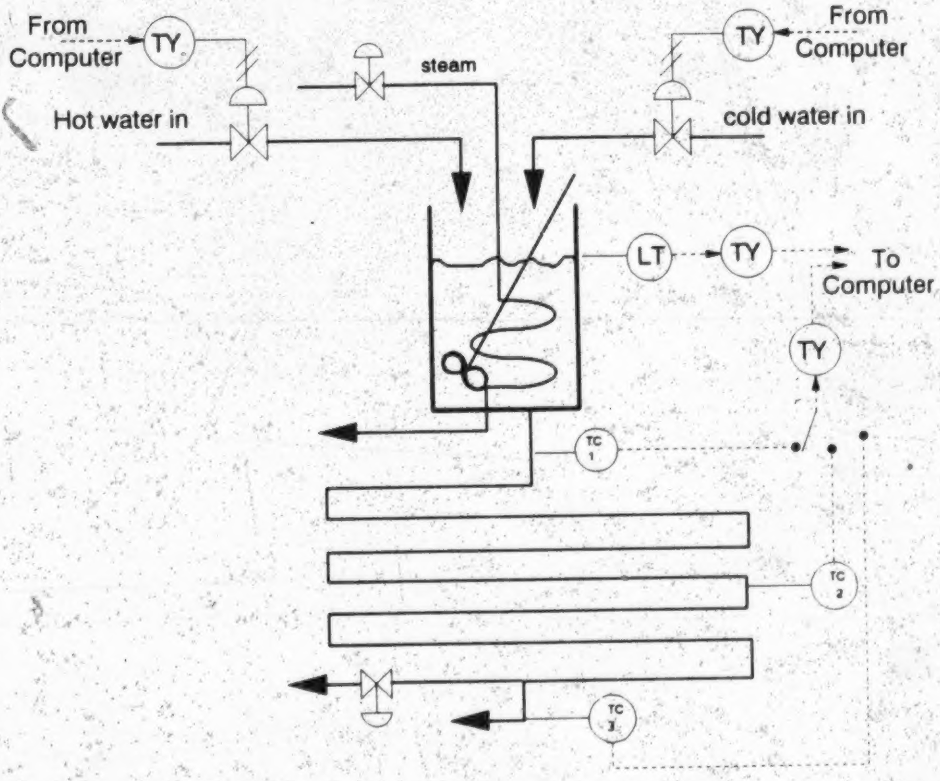

From
Computer
TY
steam
Hot water in
TY
From
Computer
cold water in
LT
TY
To
Computer
TY
TC
1
TC
2
TC
3

Figure 5.6: Schematic diagram of stirred tank heater for MIMO configuration

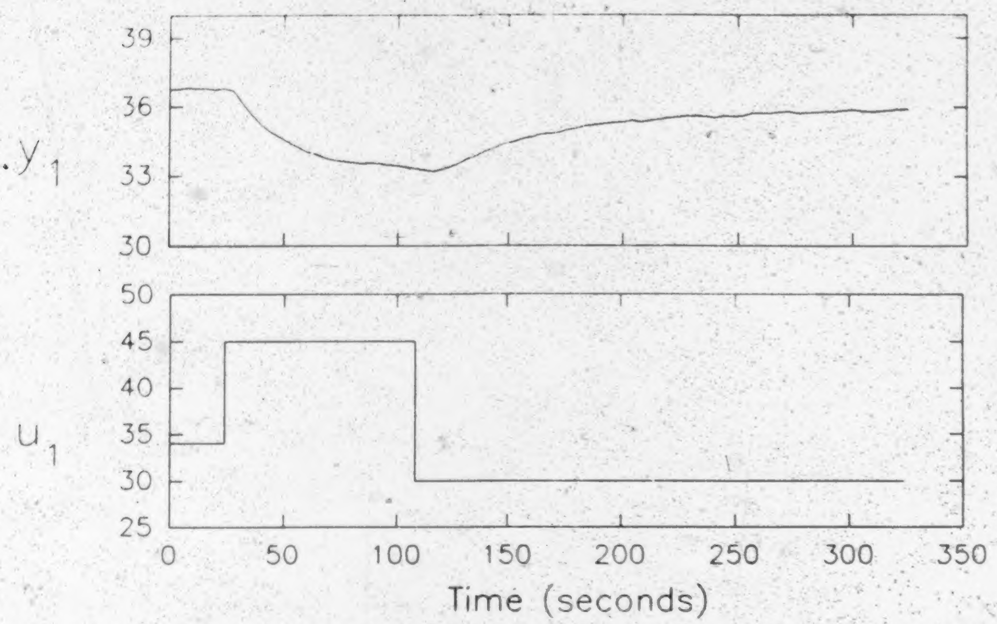


Figure 5.7: Open loop response of the temperature (^{0}C)as output (y_1) with percentage cold water valve opening (u_1) as input for the tank in MIMO configuration

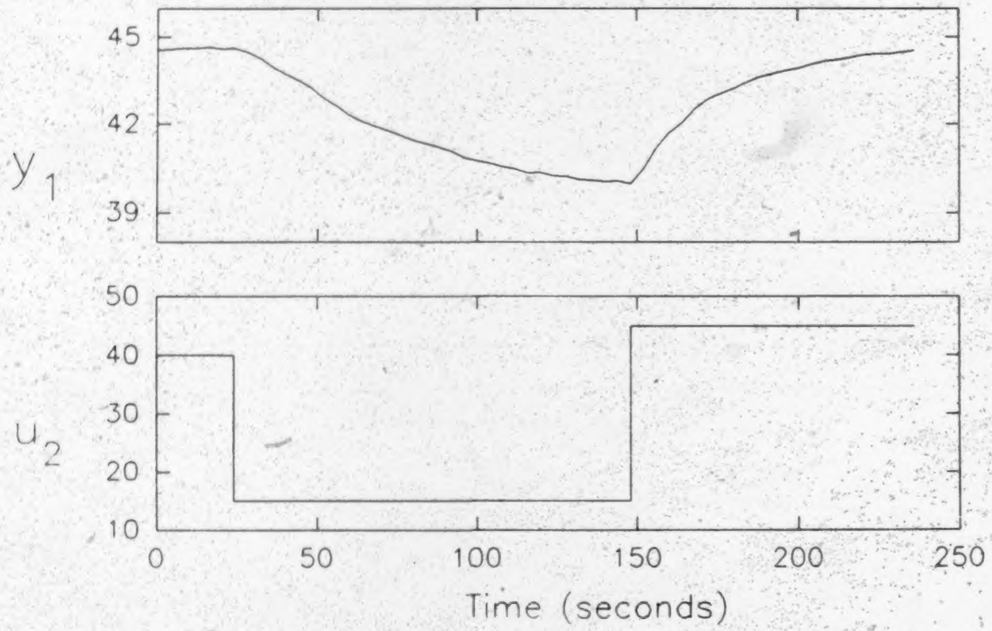


Figure 5.8: Open loop response of the temperature (^{0}C) as output (y_1) with percentage hot water valve opening as input (u_2) for the tank in MIMO configuration

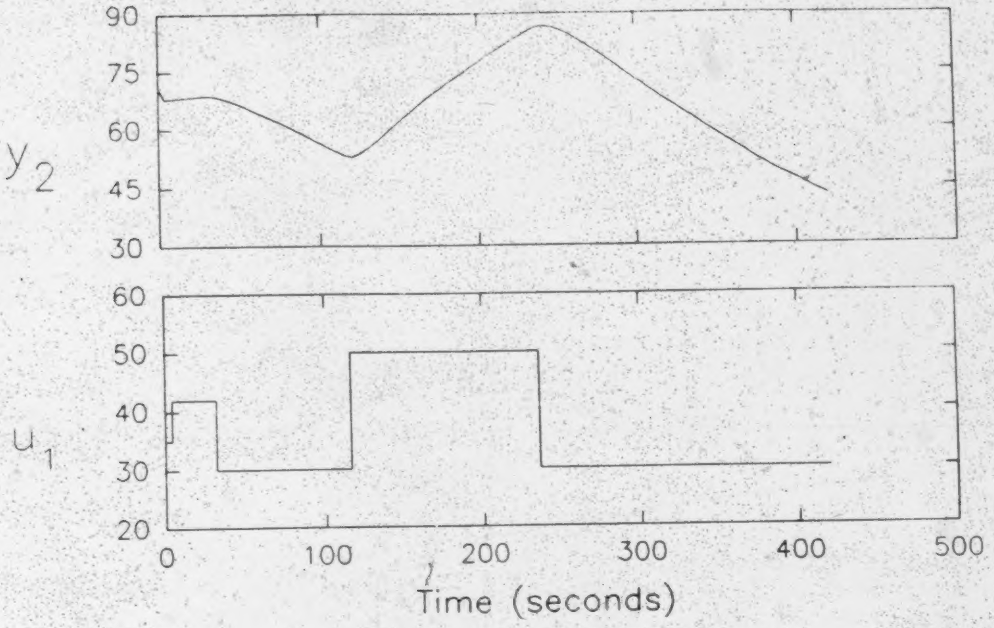


Figure 5.9: Open loop response of the pecentage tank level (y_2) as output with percent cold water valve opening (u_1) as input for the tank in MIMO configuration

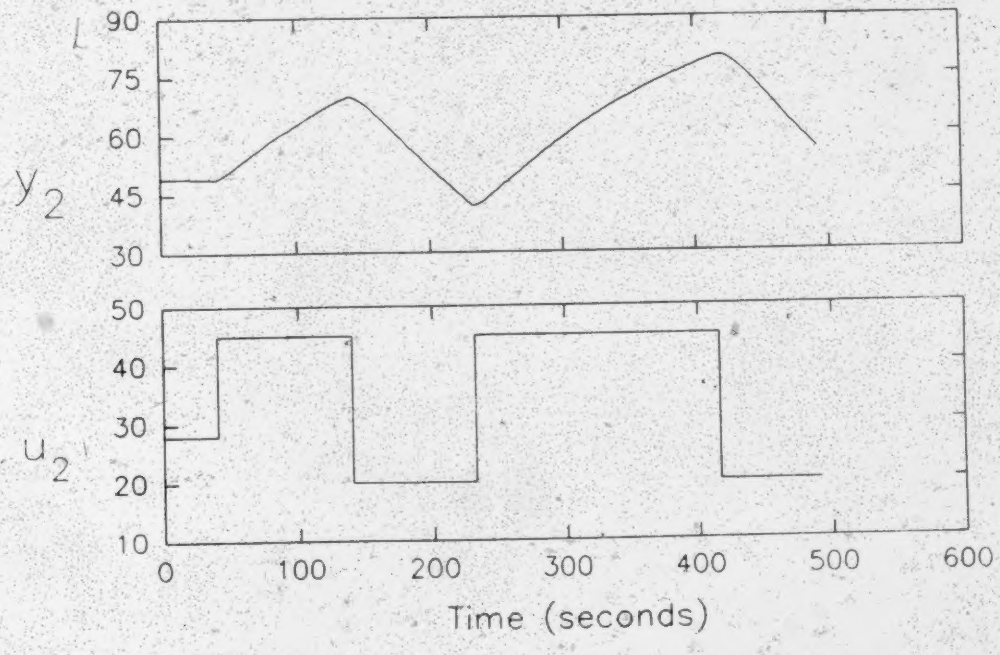


Figure 5.10· Open loop response of percentage tank level (y_2) as output with percentage hot water valve opening (u_2) as input for the tank in MIMO configuration

- y_2 is level of water in tank as percentage of maximum height of the tank,

- u_1 is percentage cold water valve opening, and

- u_2 is percentage hot water valve opening.

For the experiments performed with thermocouple # 2 shown in figure 5.6, the following transfer function matrix model was used by the controller:

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \frac{-0.35e^{-32s}}{40s+1} & \frac{0.45e^{-28s}}{44s+1} \\ \frac{.031}{s} & \frac{.0223}{s} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{5.2}$$

The open loop response of the temperature output with %cold water valve and % hot water valve opening as inputs is shown in figures 5.11 and 5.12. Notice that the two time-constants for temperature output with percent cold water valve opening and percent hot water valve opening are different because the cooling heat transfer dynamics are faster than the heating dynamics (*i.e.* heat losses through the tank and pipes account for the extra cooling). The time-delays should be identical. The fact that they are in error by one sampling interval is due to error in the empirical step-response identification.

The dynamics of the water level in tank (y_2) does not change with the distance of thermocouple in the copper tube.

The models of the process defined in equations 5.1 and 5.2 were both determined from step response data of the process. It is observed that the gain of the transfer function of water outlet temperature to hot water valve opening is substantially different for the two models. Though this may look surprising, it is possible to have such different gains for this process at different operating conditions. Material and energy balance of the process can explain the different gains of this process. Defining variables:

f_h	as inlet hot water flow rate in liters/sec
f_c	as inlet cold water flow rate in liters/sec
T	as temperature of exiting water in degrees C

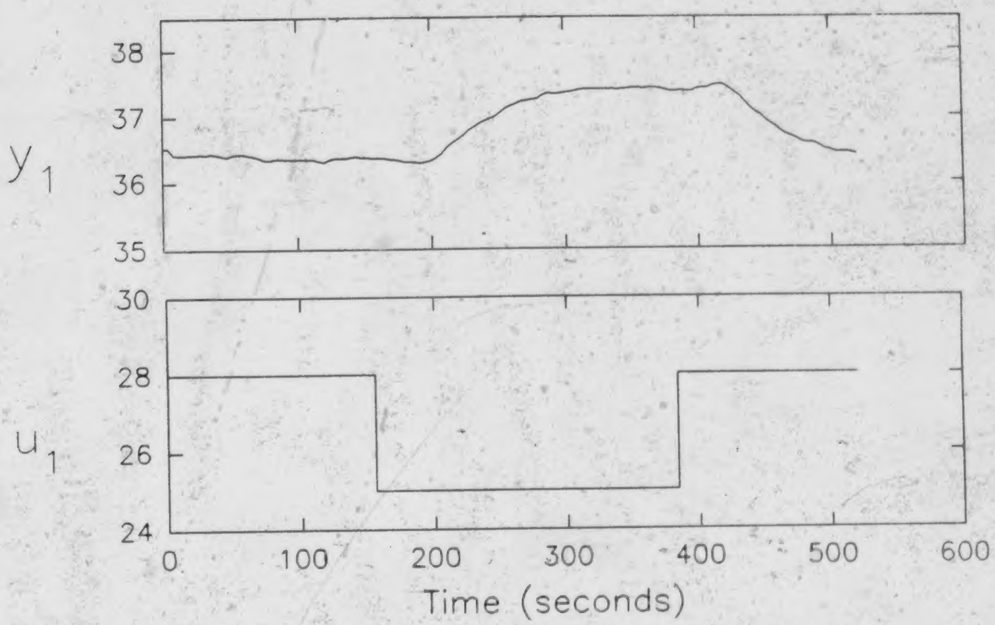


Figure 5.11: Open loop response of temperature (^{0}C) as output (y_1) at thermocouple # 2 as output with percentage cold water valve opening (u_1) as input for the tank in MIMO configuration

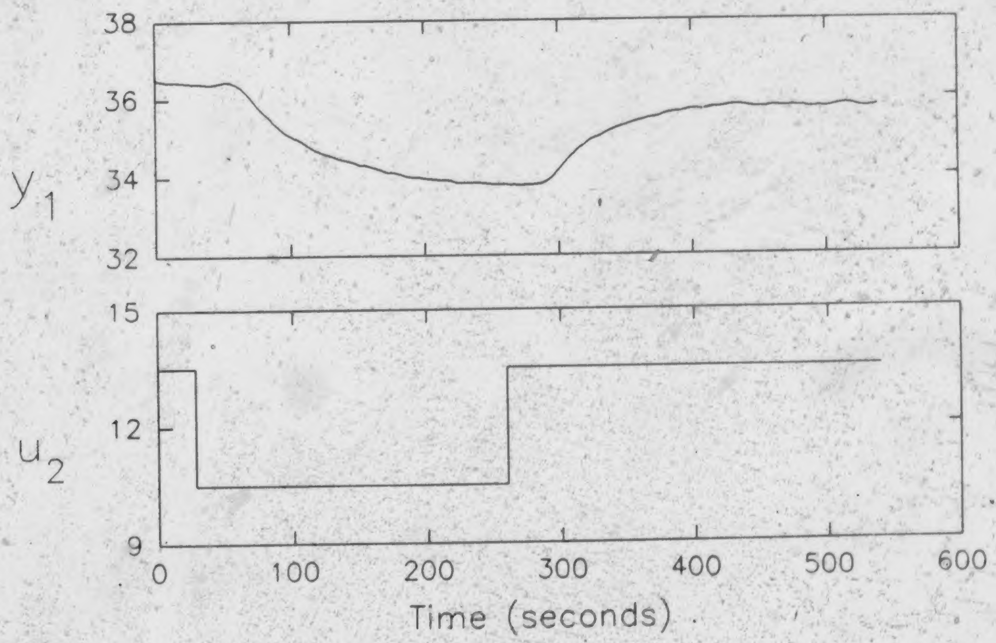


Figure 5.12: Open loop response of temperature (^{0}C) as output (y_1) at thermocouple # 2 as output with percentage hot water valve opening (u_2) as input for the tank in MIMO configuration

As has been stated earlier, the inlet hot water and cold water are at temperatures of 50 and 5°C respectively. Therefore, from mass and energy balance, the steady state temperature of the resultant exit water, assuming constant heat capacity of the water, is given by the equation:

$$T = \frac{50f_h + 5f_c}{f_h + f_c} \tag{5.3}$$

The steady state gain of the system with temperature as output and hot water flow rate as input can be found by generating a step change in hot water flow rate and measuring the change in temperature. Mathematically, this can be defined as $\partial T / \partial f_h$, at the operating point. By examining the characteristics of $\partial T / \partial f_h$, the range of linearity of the system can be examined.

$$\frac{\partial T}{\partial f_h} = \frac{45f_c}{(f_h + f_c)^2} \tag{5.4}$$

As can be seen from the above equation, the steady state gain of the temperature-hot water system given by $\partial T / \partial f_h$ is non-linear and is strongly dependent on f_h and f_c. The plot of the $\partial T/\partial f_h$ is shown in figure 5.13, for different values of f_c. Each point on the y-axis shows a gain of the process for different operating points with the x-axis representing f_h and each line in the plot represents different f_c. The gain of the process changes significantly for different operating conditions. This figure illustrates the fact that the process is highly non-linear with respect to the temperature as output.

The gains for the level output (y_2) are fairly constant. They are a function of height of water in the tank and the resistance offered by the long copper tube to the exiting water. Since these parameters are fairly constant, the level output is well modeled by pure integrators for both the inputs.

To analyze the amount of coupling in the process, consider Bristol's relative gain array (RGA) for the transfer function matrices of the process. The RGA for the transfer

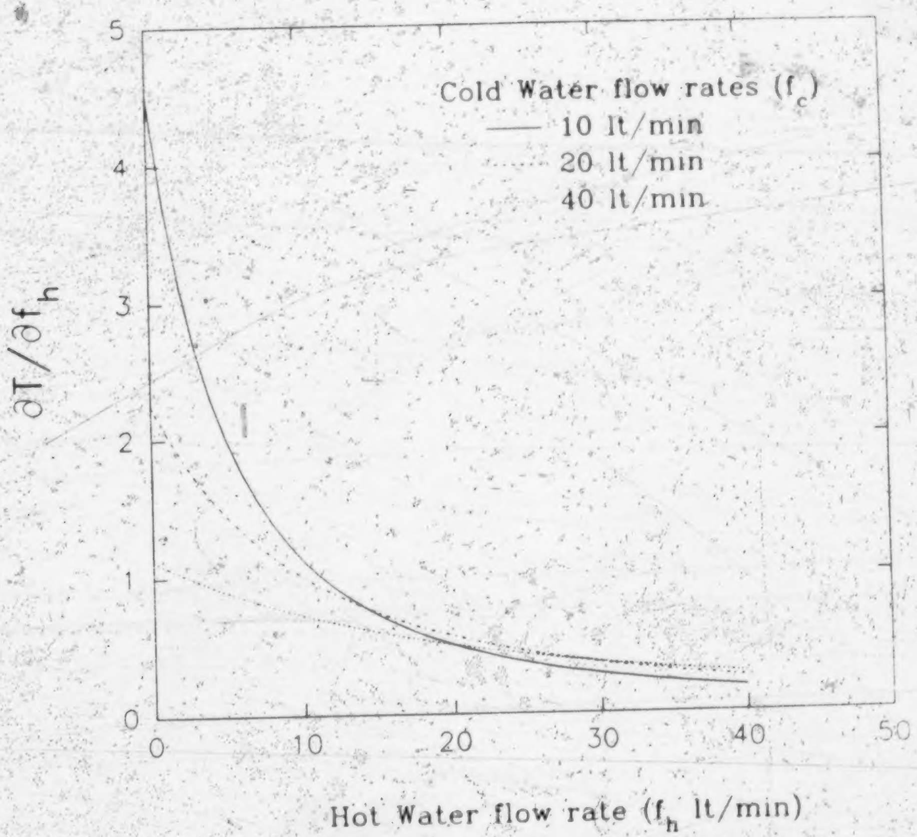


Figure 5.13: The non linear characteristics of the MIMO experimental equipment

function matrix of equation 5.1 is:

$$\Lambda_1 = \begin{bmatrix} 0.580 & 0.420 \\ 0.420 & 0.580 \end{bmatrix} \tag{5.5}$$

and the RGA for the transfer function matrix in equation 5.2 is:

$$\Lambda_2 = \begin{bmatrix} 0.355 & 0.644 \\ 0.644 & 0.355 \end{bmatrix} \tag{5.6}$$

The RGA's in equation 5.5 and 5.6 suggest different diagonally dominant systems under different operating conditions. Also note that the elements of the RGA matrices are close to 0.5 and which implies that the system is highly interacting.

This highly non-linear, interacting, and open loop unstable system is a challenging process to control. The sampling interval was chosen as 4 seconds for all the experimental runs for MIMO configuration. As the MIMO system is open loop unstable, the sampling time was decreased when compared with the SISO experimental runs.

5.3.2 Simulation of the MIMO Stirred-tank Process

This section simulates the stirred tank mixer described in section 5.3.1. The simulations were conducted on the process described by the transfer function matrix of equation 5.1.

Two simulations are presented in this section to show the effect of maximum control horizon, output weighting, and rate constraints. The discretized model with a sampling time of 4 seconds was used for simulations. The first simulation was performed with MGPC tuning parameters set to: $N_1 = 1$, $NU = 2$, $\lambda_1 = \lambda_2 = .05$, and $y_{wt1} = y_{wt2} = 1$, $\{\Delta u_1 = \Delta u_2\}_{max} = 100$ and $\{\Delta u_1 = \Delta u_2\}_{min} = -100$. For practical purposes, the rate constraints were absent as all of the calculated control moves were less than 100 units in magnitude. The $T(q^{-1})$ polynomial was selected to be $1 - 0.8q^{-1}$. The maximum control horizon N_2 is 15 for the initial 225 sampling intervals and then was changed to 5 for the remaining part of the simulation. The time-trajectory plot for these settings is shown in figure 5.14. This simulation is presented for comparison with the experimental runs in section 5.3.3.

The second simulation evaluates the performance of the controller for different output weighting (y_{wt}) and rate constraints. The tuning parameters of the controller were set to $N_1 = 1$, $N_2 = 15$, $NU = 2$, $\lambda_1 = \lambda_2 = .05$, and $y_{wt1} = 1$. The $T(q^{-1})$ was selected to be $1 - 0.8q^{-1}$.The simulation was started with $y_{wt2} = 1$ and rate constraints $\{\Delta u_1 = \Delta u_2\}_{max} = 100$ and $\{\Delta u_1 = \Delta u_2\}_{min} = -100$. The simulation of this run is shown in figure 5.15. At sampling interval 225, y_{wt2} was changed to 0.3. It is observed that the closed loop system was becoming unstable. Rate constraints were implemented at sampling interval 275 as: $\{\Delta u_1 = \Delta u_2\}_{max} = 5$ and $\{\Delta u_1 = \Delta u_2\}_{min} = -5$. It can be seen in figure 5.15 that the system becomes stable and performs satisfactorily. This clearly demonstrates that a system can become unstable for some values of output weighting and that rate constraints can stabilize some systems which would be otherwise unstable.

The above simulations are presented for comparison with some of the experimental runs described in next section.

5.3.3 MIMO Experimental Runs on Stirred Tank

The MGPC was implemented with the models described by equation 5.1 and 5.2 for thermocouples #1 and #2 respectively. The models of the process were identified off-line. As stated in section 5.3.1, this is a difficult process to control because of the instability of the open loop process and the highly non-linear characteristics of the process. This process cannot be well represented by a fixed parameter linear model and hence there is a large amount of model-plant mismatch. Due to this, the controller tuning parameters were set to conservative 'non-default' values. Unlike the simulation of the process represented in section 5.3.2, where there was no model process mismatch, tuning the controller for the experimental runs was found to be fairly time-consuming.

This section is organized as follows: Section 5.3.3.1 demonstrates that a tuned

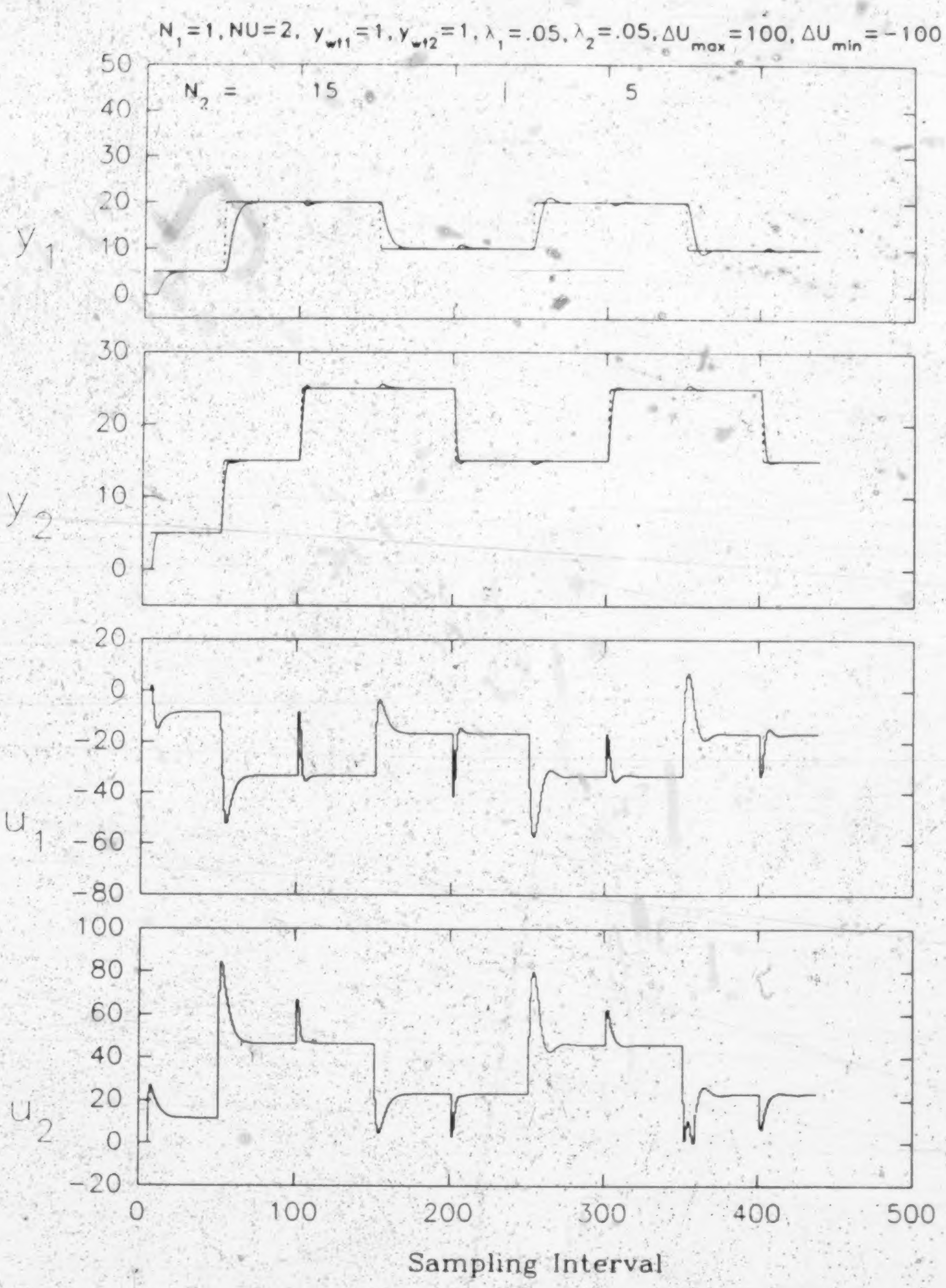


Figure 5.14: Simulation of the 2 × 2 stirred tank process for different output prediction horizons

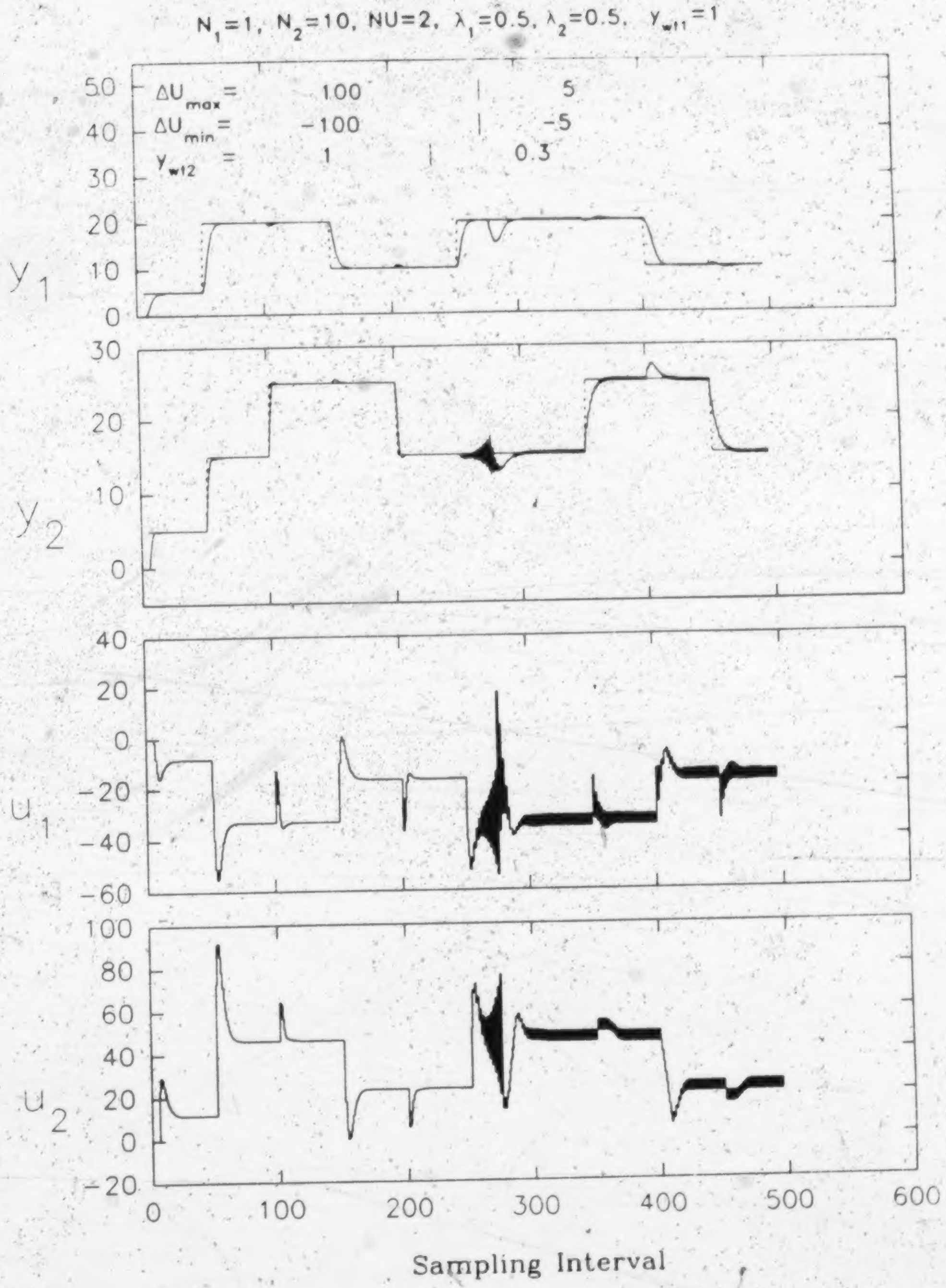


Figure 5.15: Simulation of the 2 × 2 stirred tank process for evaluating the effect of output weighting and rate constraints

MGPC can control this challenging process, section 5.3.3.2 compares the performance of constrained and unconstrained MGPC, sections 5.3.3.3 and 5.3.3.4 demonstrate the effect of variation of maximum prediction horizon and the effect of variation of output weighting respectively, section 5.3.3.5 presents the performance of a well tuned MGPC for thermocouple # 2 of the stirred tank in figure 5.6 and section 5.3.3.6 demonstrates the effect of variation of control weighting. An experimental run was also conducted with multiloop PID controllers to compare its performance against MGPC and is described in section 5.3.3.7.

5.3.3.1 Performance of a Tuned MGPC

This run demonstrates that the non-linear, open loop unstable process can be controlled by multivariable GPC. The values of the various tuning parameters of the MGPC were found by trial and error. The values so obtained, for a well controlled process, were $N_1 = 1$, $N_2 = 5$, $NU = 2$, $\lambda_1 = 1.5$, $\lambda_2 = 1.5$, $y_{wt1} = 1.0$, $y_{wt2} = 0.2$. No constraints were imposed on the rate of change of manipulated variables. The controller was turned on at the sampling interval 5. Step disturbances were introduced in the process by closing the steam valve from 5% to 3%[1] at sampling interval 350, and by changing the outflow rate of water from the stirred tank at sampling interval 560. The performance of the MGPC controlled process with the above tuning parameters is shown in figure 5.16.

A run was also performed with the same values of controller tuning parameters and with rate constraint on both the manipulated variables of $-2 \leq \Delta \mathbf{u} \leq +2$. The Step disturbances for this run were introduced in the process at sampling interval 350 by closing the steam valve from 5% to 3% and at sampling interval 590 by changing the exit flow-rate of water from the stirred tank. The performance of this system is shown in figure 5.17.

Figures 5.16 and 5.17 show that both the control schemes were able to track the outputs and were able to handle the step disturbances. Though the controller for each of

---

[1] the values for disturbance are so chosen because if the steam valve was opened from 0% to 2%, the condensed steam in the so far unused steam pipe line would not give the desired disturbance

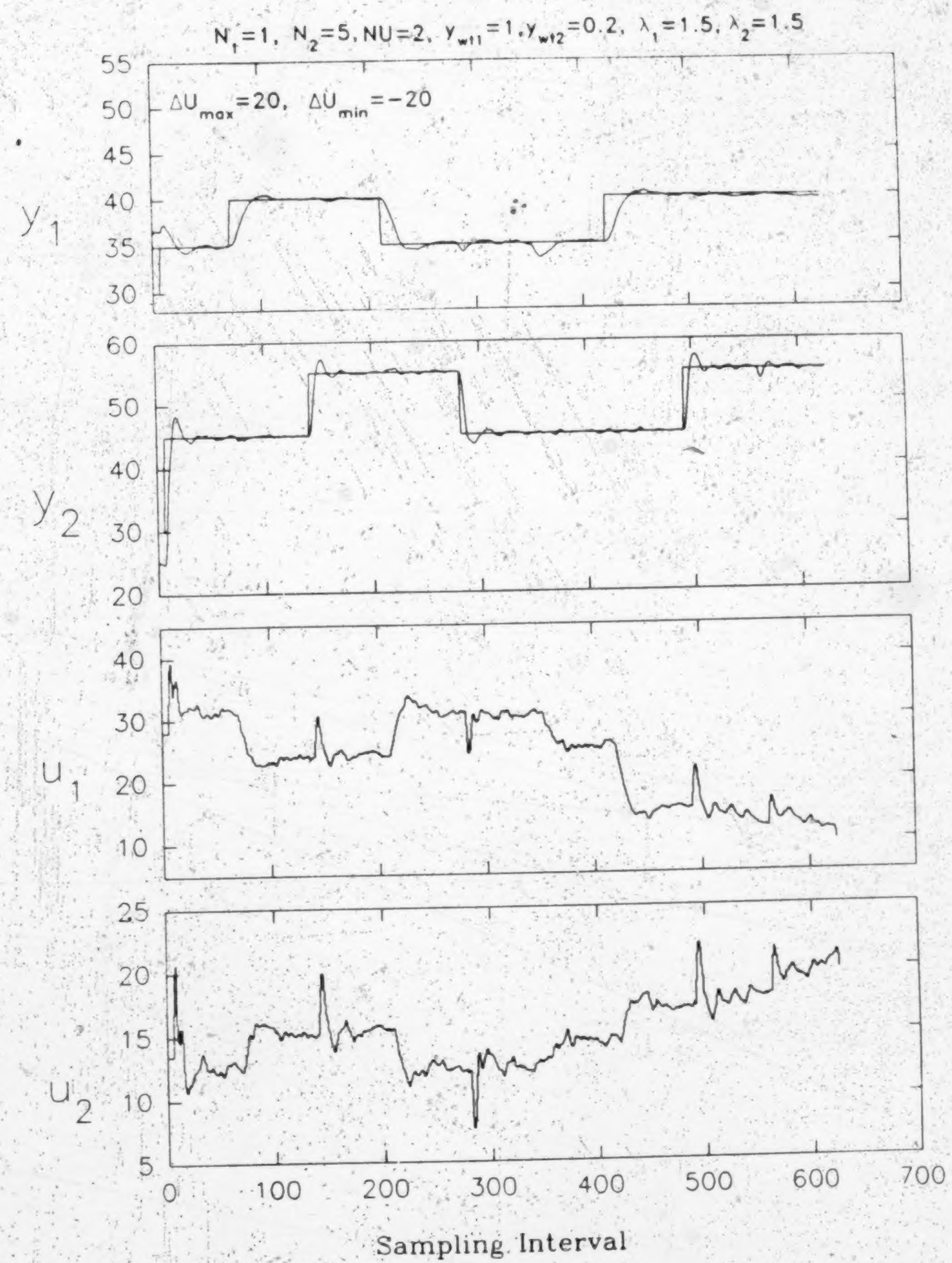


Figure 5.16: Control of stirred tank with a well tuned unconstrained MGPC

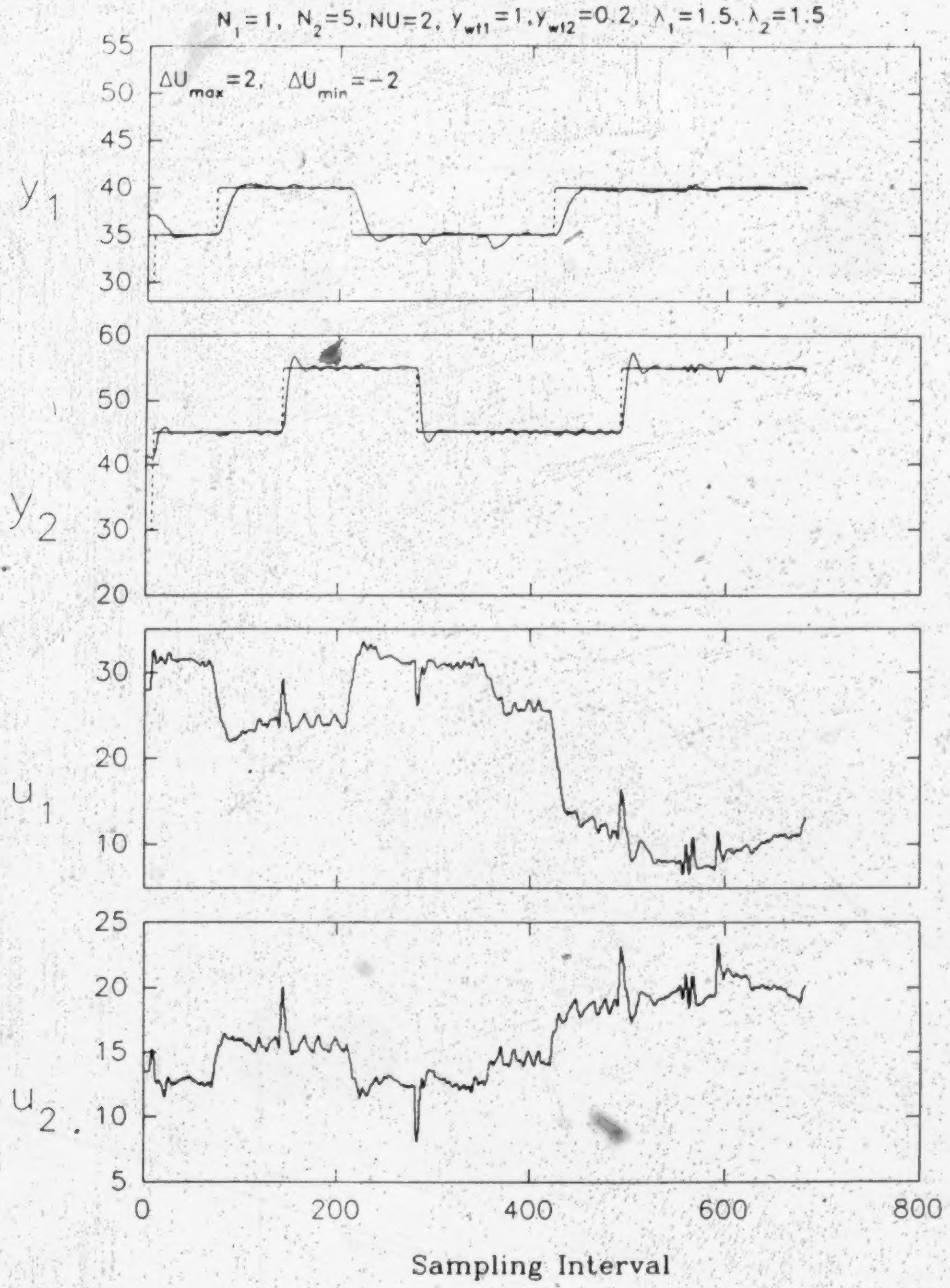


Figure 5.17: Control of stirred tank with a well tuned constrained MGPC

the run was able to minimize the interactions between the two outputs, it was not able to eliminate them. Nevertheless the interactions have been minimized significantly. Note that the values of the controller tuning parameters for the experimental runs are nowhere similar to the values used for the simulations of the process in section 5.3.2. This is due to the large amount of model process mismatch and non-linearity of the process in the experimental runs.

5.3.3.2 Performance Comparison of Constrained and Unconstrained MGPC

This run compares the performance of constrained MGPC and unconstrained MGPC. The controller was intentionally not well tuned, to explicitly illustrate the advantage of using the constrained MGPC over the unconstrained MGPC. The MGPC tuning parameters were set to: $N_1 = 1$, $N_2 = 7$, $NU = 2$, $\lambda_1 = 1.2$, $\lambda_2 = 1.0$, $y_{wt1} = 1.0$, $y_{wt2} = 0.25$ The controller was started with these settings at sampling interval 5 with rate constraint on both the manipulated variables of $-2 \leq \Delta \mathbf{u} \leq +2$. After 200 sampling intervals, the rate constraints on both the manipulated variables were changed to $-1 \leq \Delta \mathbf{u} \leq +1$. At sampling instance 400, the rate constraints on the controller were withdrawn. The performance of this run is shown in figure 5.18.

The figure shows that the controller was able to control the process in the presence of the rate constraints on the controller. When the rate constraints were tightened at sampling interval 200, the ringing action of the manipulated variables was reduced. However, the setpoint tracking was slower than before. Relaxation of the rate constraints at sampling interval 400 on the manipulated variables resulted in the closed loop system being unstable. From this run we can conclude that rate constraints can stabilize some of the unstable systems. Another experimental run in subsequent section also supports the same hypothesis. It can also be inferred that rate constraint can slow down the dynamics of the systems.

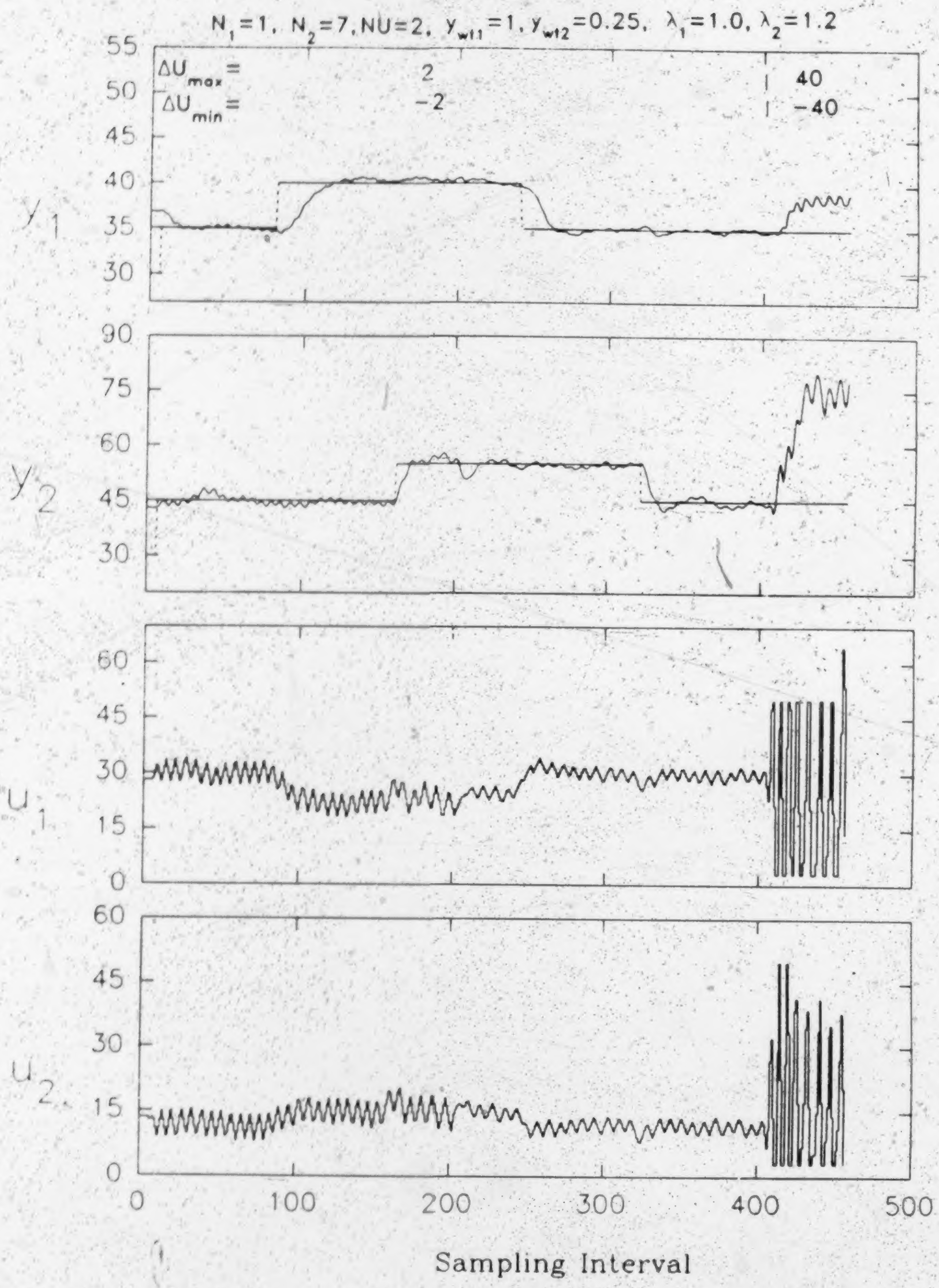


Figure 5.18: Effect of constraints on the control of stirred tank with MGPC

5.3.3.3 Effect of Variation of Maximum Prediction Horizon

This run emphasizes the importance of selecting appropriate N_2 for processes with large model plant mismatch. The MGPC tuning parameters for this experimental run were set to: $N_1 = 1$, $NU = 2$, $\lambda_1 = 1.5$, $\lambda_2 = 1.5$, $y_{wt1} = 1.0$, $y_{wt2} = 0.2$. These settings are the same as those of the well tuned MGPC in section 5.3.3.1. The controller was started at sampling interval 5, with $N_2 = 2$. N_2 was changed to 4, 6 and 8 at sampling intervals 200, 350 and 480 respectively. Rate constraint of $-2 \leq \Delta \mathbf{u} \leq +2$ was imposed on both the manipulated variables, on the hitherto unconstrained MGPC, at sampling interval 500. The performance of this run is shown in figure 5.19.

For control horizon, N_2, equal to 2, the performance of the control system is poor. For control $N_2 = 4$, the performance of the system is good ($N_2 = 4$ setting makes this system the same as the well tuned GPC controller of section 5.3.3.1). For $N_2 = 6$, the controller is aggressive and has poor setpoint tracking characteristics. For $N_2 = 8$, the closed loop system is unstable. Imposing of rate constraints at sampling interval 500 on the on the manipulated variables stabilized the otherwise unstable system.

From this run, it is seen that selection of N_2 is not a trivial matter for systems with substantial model process mismatch. Selecting a small N_2 can result in poor control performance (more like generalized minimum variance control). This process being non-linear, there is a large model process mismatch. For larger maximum control horizon, N_2, the model prediction error is large and misleads the controller into calculating unwarranted control signals thereby driving the closed loop process to instability. Larger N_2 can be selected for processes with relatively small model process mismatch. Rate constraint on the manipulated variables can stabilize the unstable process, by restricting the implemented control signal.

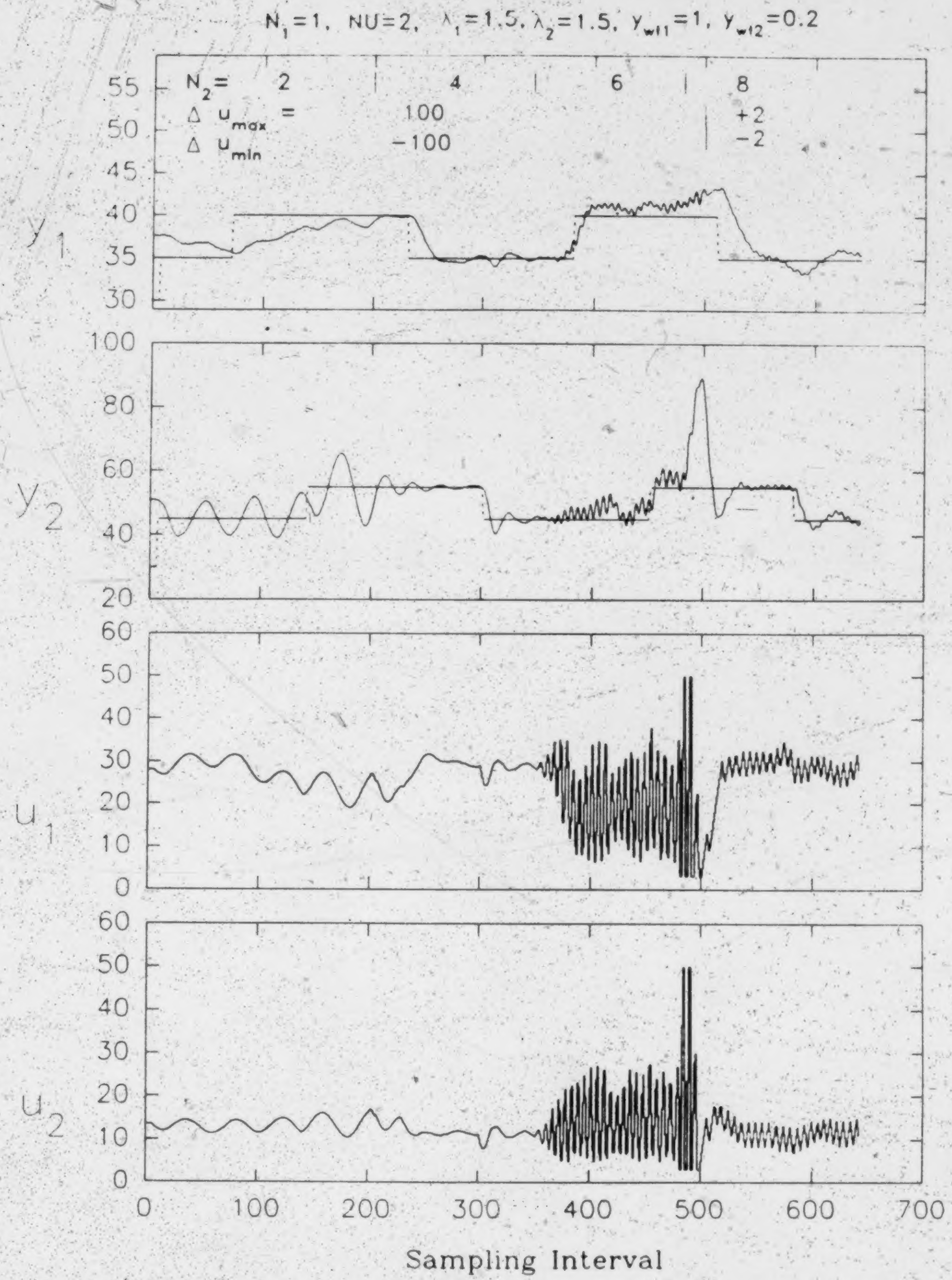


Figure 5.19: Effect of variation in maximum prediction horizon in MGPC on the control of stirred tank

5.3.3.4 Effect of variation of Output Weighting y_{wt}

MGPC has an additional tuning parameter to SISO GPC: the output weighting (y_{wt}). In MGPC, the control algorithm can weight the error in different outputs of the process by this tuning parameter. SISO processes have only one output and hence y_{wt} is irrelevant. As will be clear from the experimental runs, this is an important tuning parameter and can dramatically influence the performance of the system. The experimental runs also support the statement in section 4.2.4.4 (page 83) that larger y_{wt} for outputs with larger model process mismatch can improve the performance of the closed loop system. In the present process, the temperature (output y_1) channel has relatively larger model process mismatch when compared with level output (y_2).

The MGPC tuning parameters where set to $N_1 = 1$, $N_2 = 5$, $NU = 2$, $\lambda_1 = 1.0$, $\lambda_2 = 1.2$, $y_{wt1} = 1.0$. This run was conducted with the fixed parameter model given by equation 5.1. Since the ratio of the output weightings and not their absolute value is important, the output weighting (y_{wt1}) on temperature output was fixed equal to 1 and the output weighting (y_{wt2}) on the level output was varied. The controller was started at sampling interval 10 with $y_{wt2} = 0.01$ The output weighting y_{wt2} was varied at sampling intervals 120, 250, 370 and 450 to 0.1, 0.2, 0.3, 0.5. The performance of this run is shown in figure 5.20.

The tracking of setpoint for level is poor for $y_{wt2} = 0.01$. The MGPC objective function gives a weight of only 1% to control the error in the level output when compared to the weight it gives to control the error in the temperature output. The performance of the system improves substantially for $y_{wt2} = 0.1$. The performance of the closed loop system is good for $y_{wt2} = 0.2$. The controller starts loosing control over the process with $y_{wt2} = 0.3$ and is out of control for $y_{wt2} = 0.5$.

The output y_1 (temperature) has much larger model process mismatch than output y_2 (level), as illustrated in the section 5.3.1. The ratio of y_{wt1} to y_{wt2} was greater than

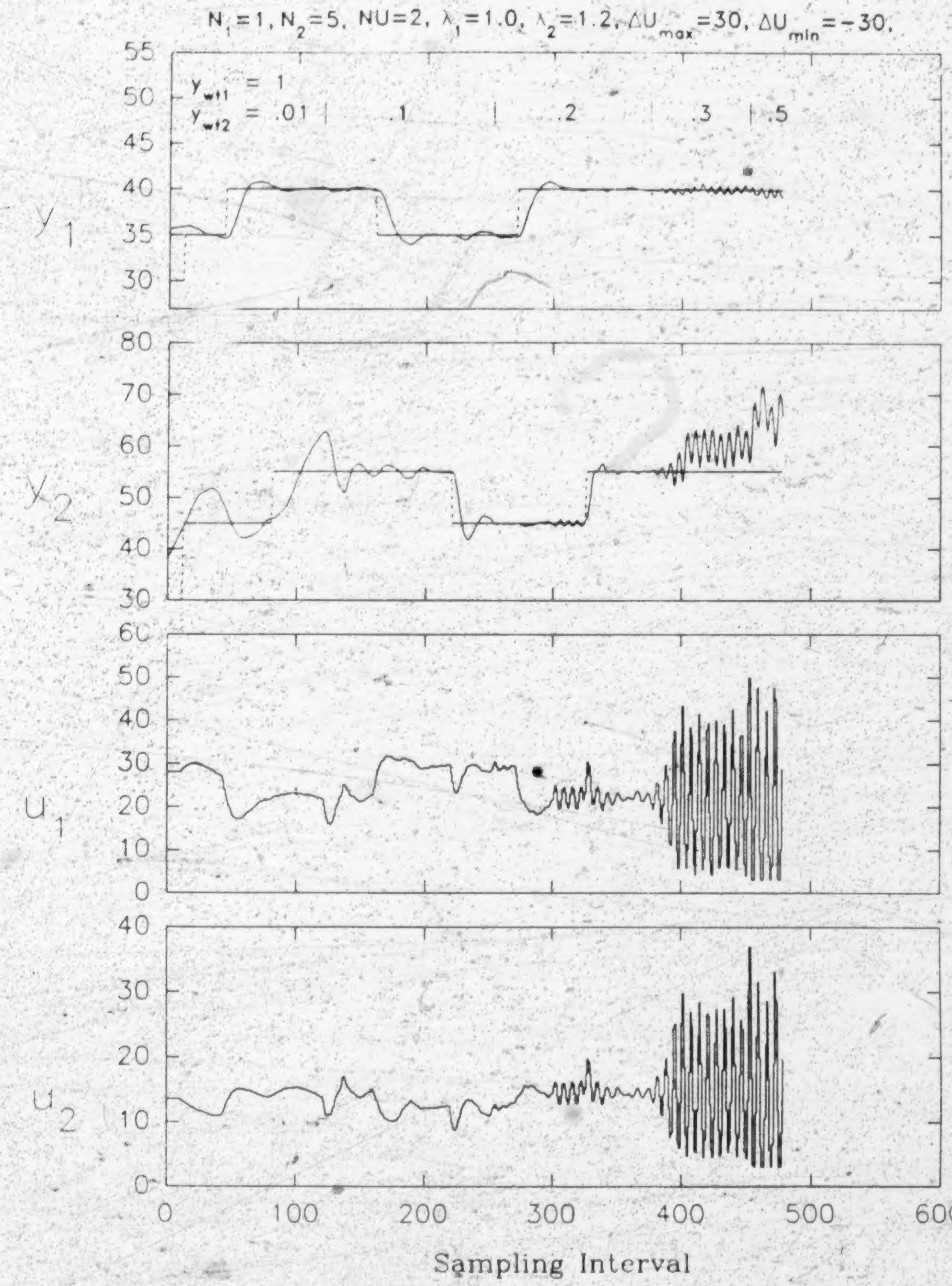


Figure 5.20: Effect of variation in output weighting in MGPC on the control of stirred tank

1 for achieving closed-loop stable system. A very large ratio, when y_{wt2} was 0.01, also is not desirable as it results in poor control of one of the outputs. This run shows that the proper selection of the tuning parameter y_{wt} is crucial for multivariable processes in achieving stable and desirable control. This experimental run also supports the statement in section 4.2.4.4 (page 83) that larger output weighting is required for outputs with larger model process mismatch for better control.

5.3.3.5 Performance Of The MGPC for Thermocouple # 2

This run was performed with the temperature measured at thermocouple # 2 (as shown in figure 5.6). The fixed parameter model described by equation 5.2 was used by the MGPC algorithm. The tuning parameters were tuned for this run by trial and error. The MGPC tuning parameters were set to: $N_1 = 1$, $N_2 = 10$, $NU = 2$, $\lambda_1 = 3.0$, $\lambda_2 = 3.5$, $y_{wt1} = 1.0$, $y_{wt2} = 0.05$. No constraints were imposed on the rate of change of manipulated variables. The controller was turned on at the sampling interval 5. A step disturbances was introduced in the process by closing the steam valve from 5% to 3%, at sampling interval 400, and by changing the outflow rate of water from the stirred tank, at sampling interval 675. The performance of the MGPC controlled process with the above tuning parameters is shown in figure 5.21. The oscillatory behavior of the y_2 for the initial 200 sampling intervals was due to the hot water supply temperature, from the utilities, drifting from 40 ^{0}C to 50 ^{0}C. By further detuning the controller this oscillatory behavior of the y_2 output can be avoided.

As can be seen from the performance of this system in figure 5.21, it is able to track the setpoint changes and reject the step disturbances in both the outputs. This run also proves that the MGPC controller can handle multivariable systems with unequal amount of delays in different channels of the process.

The program written for implementing MGPC control was with equal values of N_1 and N_2 for all the output channels. While conducting this experiment, it was felt that a

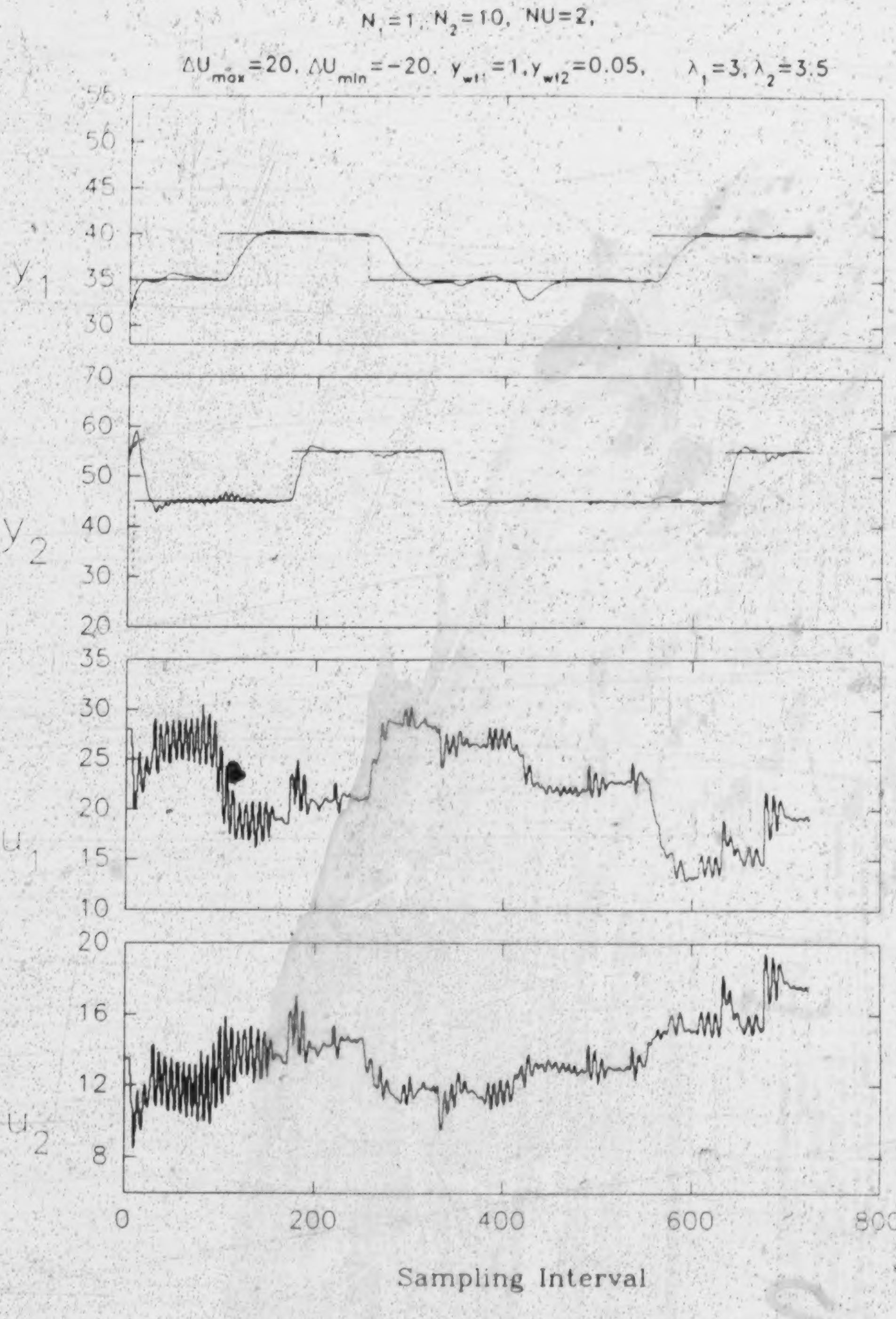


Figure 5.21: Control Of stirred tank for temperature measured at thermocouple # 2 with MGPC

better control performance could have been achieved if N_1 and N_2 could have been varied separately for different output channels. This is a limitation of the current control software (which can be removed by modifying a part of the software) and not of the MGPC algorithm.

5.3.3.6 Effect Of Variation of Control Weighting

This run was performed to demonstrate the effect of variation of control weighting (Λ) on the performance of the system. This run was performed with temperature measured from thermocouple # 2, schematically shown in 5.6, and with the model described by equation 5.2.

For this run, the tuning parameters of the controller were set to: $N_1 = 1$, $N_2 = 10$, $NU = 2$, $y_{wt1} = 1.0$, $y_{wt2} = 0.05$. The controller was started at sampling interval 5 with control weighting factors $\lambda_1 = 3.5$ and $\lambda_2 = 4.5$. The control weighting parameters were changed to $\lambda_1 = 3.5$ and $\lambda_2 = 4.0$ at sampling interval 320, to $\lambda_1 = 3.5$ and $\lambda_2 = 3.5$ at sampling interval 470, to $\lambda_1 = 3.0$ and $\lambda_2 = 3.5$ at sampling interval 620, to $\lambda_1 = 2.5$ and $\lambda_2 = 3.5$ at sampling interval 770, and to $\lambda_1 = 2.5$ and $\lambda_2 = 3.0$ at sampling interval 1065. The performance of this run is shown in figure 5.22.

The control moves grow larger as the control weighting decreases and becomes unstable for $\lambda_1 = 2.5, \lambda_2 = 3.0$. Also, for $\lambda_1 = 2.5, \lambda_2 = 3.5$, for sampling intervals between 770 and 1065, the control action is 'smooth' for some time and then becomes aggressive. For these values, the process is marginally stable and may become unstable. This run suggests that the controller should be tuned with large values of λ_1 and λ_2 for stable control of non-linear systems.

5.3.3.7 Performance of MIMO System With Multiloop PID Controllers

An experimental run was conducted with multiloop PID controllers to control the MIMO system. The temperature for this run was measured at thermocouple # 1. Pairing of

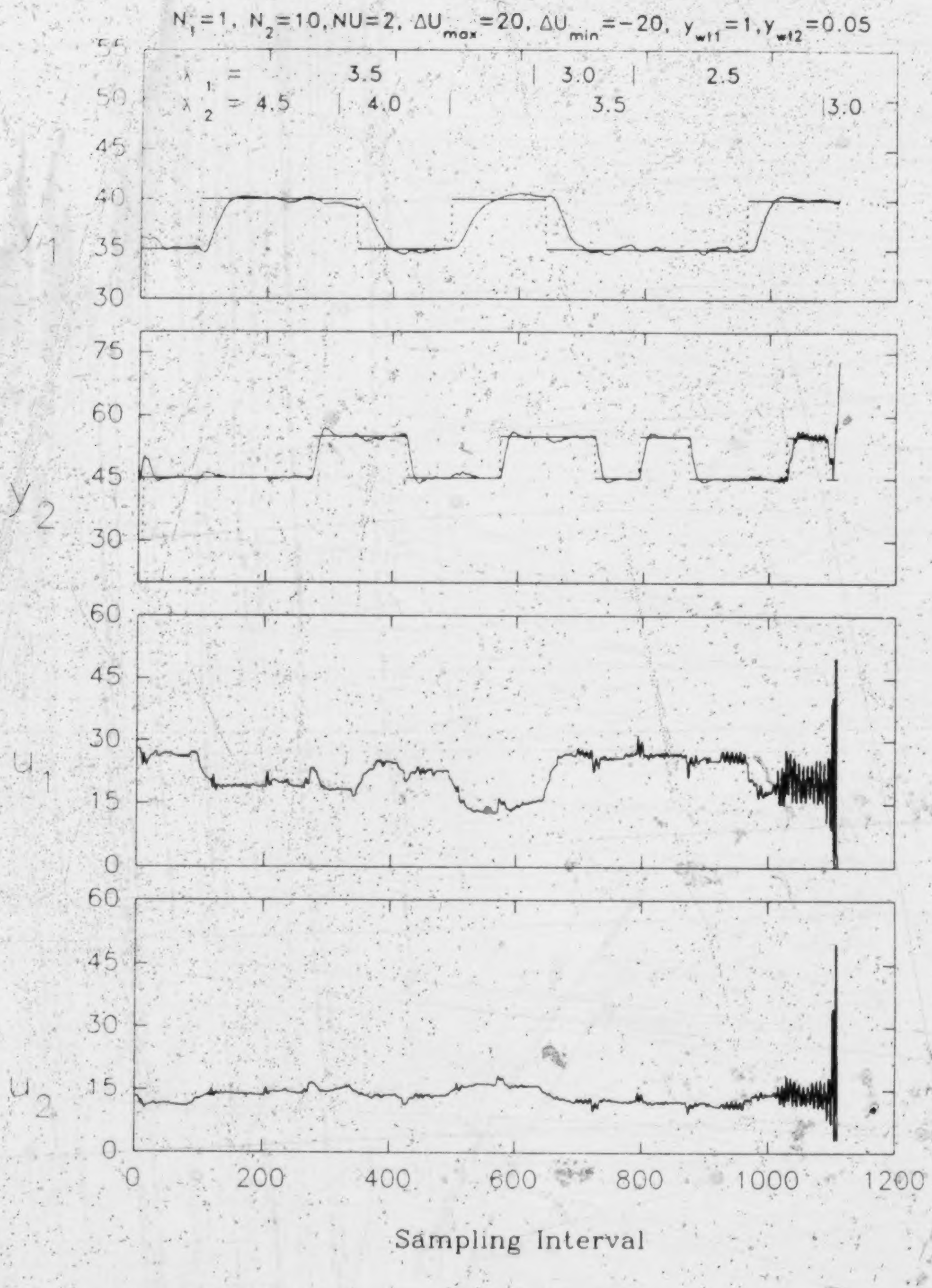


Figure 5.22: Effect of variation in control weighting in MGPC on the control of stirred tank

outputs to inputs, unlike MGPC which does not require explicit pairing of outputs and inputs, was selected on the basis of the relative gain arrays of the model in equation 5.5. The temperature control was paired with percentage valve opening of cold water and level control was paired with percentage valve opening of hot water. Bristol's relative gain array (RGA) analysis is based on steady state gain matrix of a system and is independent of dynamics and delays in the process. If the RGA based on the steady state gain matrix of the thermocouple# 2 (equation5.6) is used, then the pairing will get reversed. Note that the steady state gain matrices for thermocouples #1 and #2 are different due to modeling of the process at different operating conditions (section 5.3.1).

The tuning of the multiloop PID controllers was a tedious task. The PID constants finally implemented for controlling the temperature with percentage cold water valve opening were $K_p = -0.05$, $K_i = -0.04$, $K_d = -0.14$ and for controlling the level with percentage hot water valve opening were $K_p = 0.6$, $K_i = 0.02$, $K_d = 0.5$. Note that the large derivative gain in the controllers provide the causal action, which help in decoupling this highly interacting system. The control of the process was almost impossible without the use of derivative gains in the PID controllers. The performance of this run is shown in figure 5.23. A step disturbance was introduced in the process by closing the steam valve from 5% to 3%, at sampling interval 500 and by changing the exit rate of water from the stirred tank, at sampling interval 825.

As can be seen from figure 5.23, the performance of this control strategy is very poor. There are large interactions between the loops. Derivative gain in a PID controller reduces the oscillation of a closed loop process. In spite of the low integral gains and high derivative gains of the controllers, there are large oscillations in the outputs of this process. The performance of this system is very poor in comparison with the performance of the MGPC shown in figure 5.16.

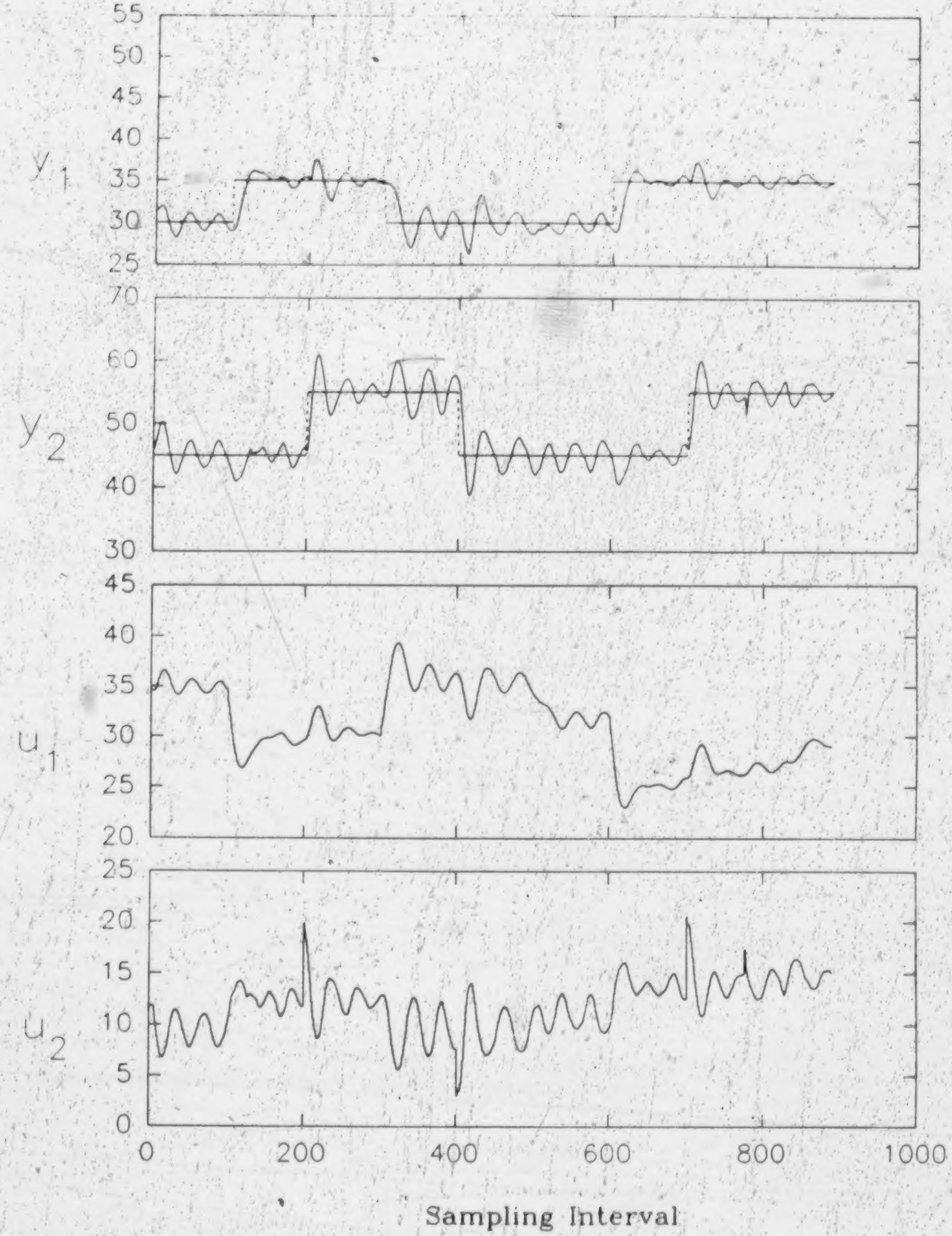


Figure 5.23: Control of stirred tank heater in MIMO configuration with tuned PID Controllers

5.4 Arterial Pressure Control with Adaptive SISO GPC

This section presents yet another illustration of the experimental evaluation of adaptive constrained and unconstrained GPC algorithm for the regulation of blood-pressure in post-operative drug therapy. The evaluation is on a fairly challenging process with varying time delays and dynamics and the fact that it works serves to show that the technique is very robust.

Post-operative drug therapy often requires blood-pressure regulation in many patients. The vasodilator, sodium nitroprusside (SNP), is routinely used in clinical practice for induced hypotension. The aim of the experiments was to control the mean arterial pressure (MAP) of a patient, in this case a dog, at a desired set-point using SNP. The experimental results demonstrate that adaptive GPC can control this challenging process. Also, this section compares the relative performances of the constrained and the unconstrained GPC algorithms. The model of the process was identified on-line using a recursive least square algorithm with a variable forgetting factor. The details of this algorithm can be found in Seborg *et al.* [45].

The experiments described in this section were performed by Kwok [25]. The runs documented here serve to show the performance of the unconstrained and constrained GPC algorithms, the latter algorithm developed by the author of this thesis. This application is being developed on dogs and will be eventually used for regulating the MAP of human patients during the post-operative drug therapy.

Subsection 5.4.1 describes the experimental equiptment used by Kwok for conducting the experiments, and subsection 5.4.2 presents the results and conclusions of the experimental runs.

5.4.1 Process Description

A schematic diagram of the experimental setup is shown in figure 5.24. A HP 78200 series monitoring system was used for measuring physiological parameters. The arterial pressure wave was measured via a carotid using a catheter pressure transducer. The measured signal was amplified in the monitoring system and both the systolic and the diastolic pressures were detected. The monitor calculates mean arterial pressure (MAP) and produces an analog voltage signal proportional to the MAP. The objective of this control exercise is to regulate this variable–MAP. The calculated mean arterial pressure was displayed on the digital board and was made available as a 0 to 3 volt analog signal corresponding to 0 to 300 mm Hg. The blood pressure signal was sampled by the Opto22 system every 10 seconds. The Opto22 converted the sampled analog signal to a 12-bit digital signal.

The computer running under QNX operating system, described in section 5.1, receives the signal from the Opto22 system via an hardware interface, RS-232. The control algorithm in the computer calculates the control signal and sends it to an IMED 929 computer enabled drug infusion pump [2]. The pump has a capacity to pump from 0 to 1599 ml/h. This process is repeated every sampling instant.

The infusion of the drug sodium nitroprusside (SNP), a vasodilator, reduces blood pressure. SNP was used to manipulate the the blood pressure of the patient. The drugs Noradrenaline (NOR) and Adenosine Monophosphate (AMP) were used to create disturbance inputs to induce hypertensive crisis (by rapidly increasing the blood-pressure) and hypotensive crisis (by rapidly reducing the blood-pressure) respectively. All the drugs were administered through the femoral vein of the dog. The experiments conducted with this equipment are described in the next subsection.

---

[2] A product developed by the IMED Corporation, San Deigo, California

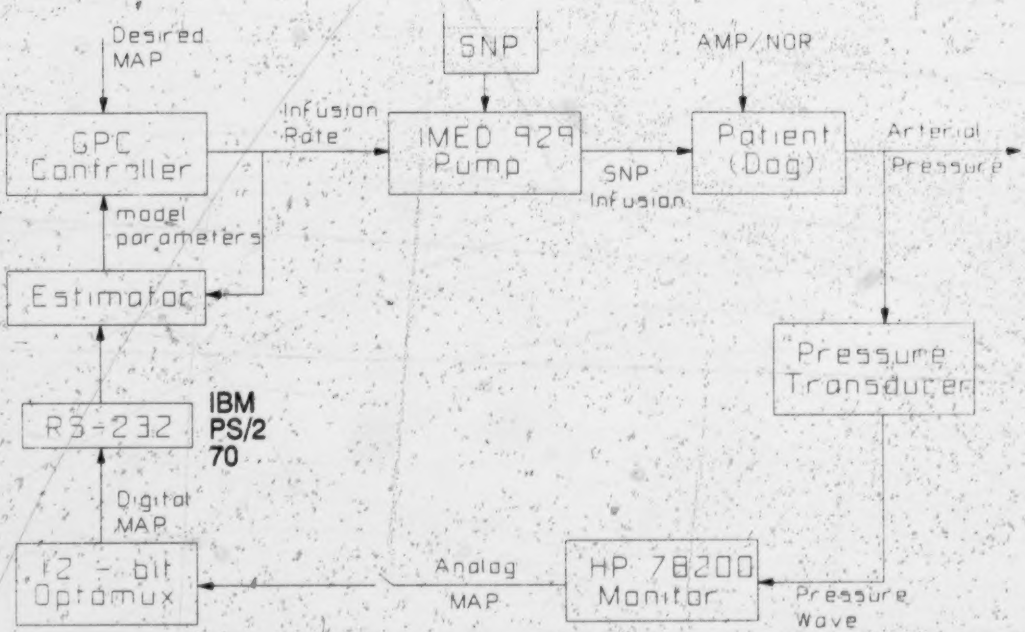

Desired MAP
SNP
AMP/NOR
GPC Controller
Infusion Rate
IMED 929 Pump
Patient (Dog)
Arterial Pressure
SNP Infusion
model parameters
Estimator
Pressure Transducer
RS-232
IBM PS/2 70
Digital MAP
12 - bit Optomux
Analog MAP
HP 78200 Monitor
Pressure Wave

Figure 5.24: Block diagram of the experimental equipment for adaptive blood pressure control

5.4.2 Experimental Results

This section describes two experimental runs conducted by Kwok on a dog in the laboratory of the Department of Pharmacology at the University of Alberta, Edmonton, Canada. The dog was anesthetized and a pulse infusion of SNP was administered to obtain the initial excitation for the adaptive algorithm.

The tuning parameters of the GPC were set to $N_1 = 2$, $N_2 = 10$, $NU = 1$, $\lambda = 0.001$ and $T(q^{-1}) = 1-0.8q^{-1}$ for both the runs. The model of process was identified by a recursive least square algorithm with variable forgetting factors. For both the runs, a second order model with a 5^{th}-order B polynomial was identified. The trajectories of 3 of the 7 parameters estimated on-line for the second order model for both the experimental runs are shown in figures 5.25 and 5.26.

The first run was conducted with the unconstrained GPC. The time trajectory plot for this run is shown in figure 5.25. The second run was conducted with GPC, having constraints on the change of the manipulated signal. The constraint was: -20 ml/hr $\leq \Delta u \leq 20$ ml/hr. The time trajectory plot for the constrained GPC run is shown in figure 5.26. Both the plots also show selected identified model parameters. All the disturbances created during the runs by Noradrenaline (NOR) and Adensone Monophosphate (AMP) are shown at the appropriate sampling interval on the time trajectory plots of both the runs.

Both the runs perform a good job on setpoint tracking. The disturbance rejection of the constrained algorithm is better than the unconstrained algorithm. For the NOR disturbance at sampling interval 375 of the unconstrained GPC run (figure 5.25), the arterial pressure of the patient drops well below the setpoint, after rejecting the disturbance. For the NOR disturbance at sampling interval 315 of the constrained GPC run (figure 5.26), the arterial pressure drop below the setpoint after rejecting the disturbance is low.

The experimental runs illustrate the applicability of constrained and unconstrained GPC for sophisticated control problems. These experimental runs also demonstrate that

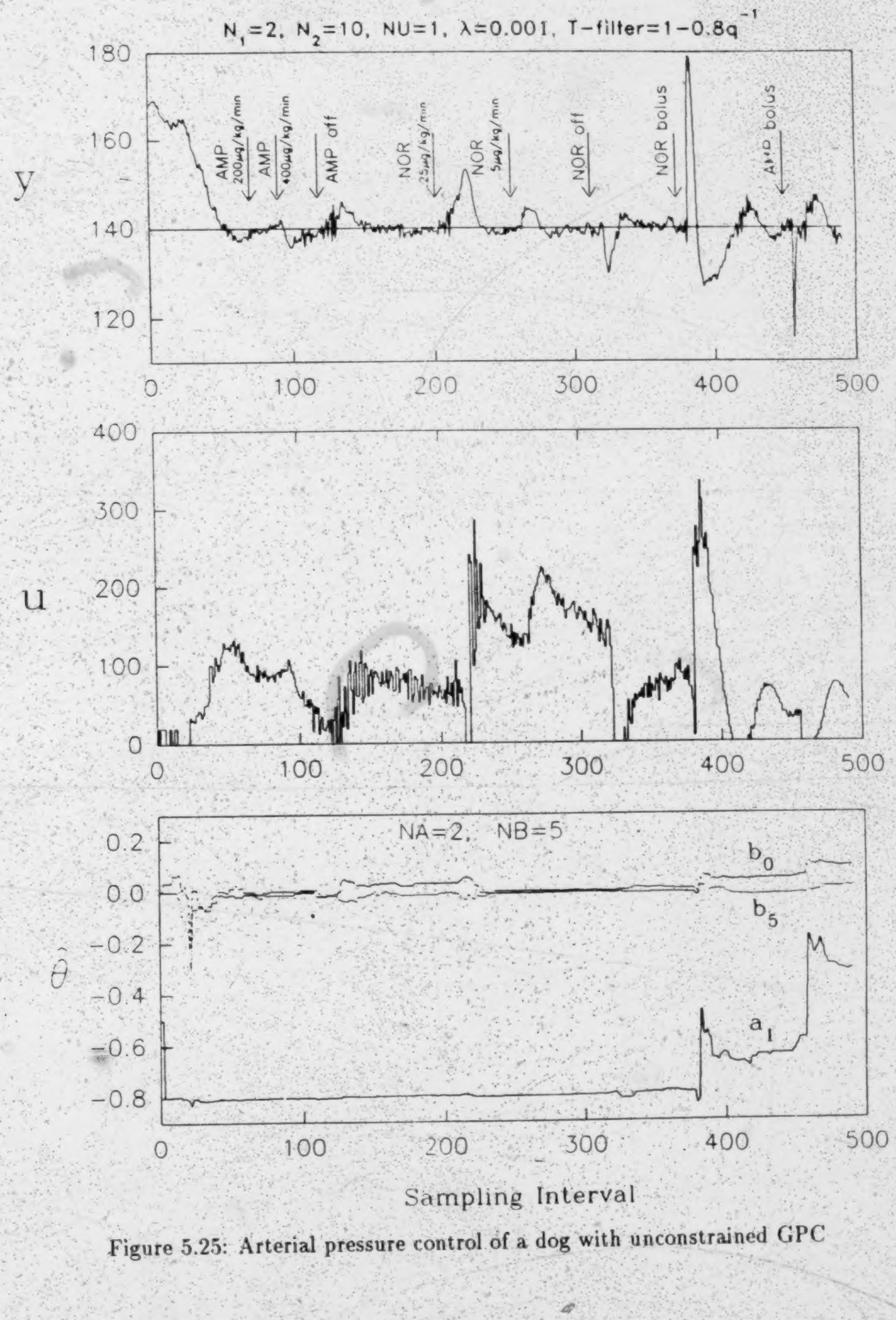


Figure 5.25: Arterial pressure control of a dog with unconstrained GPC

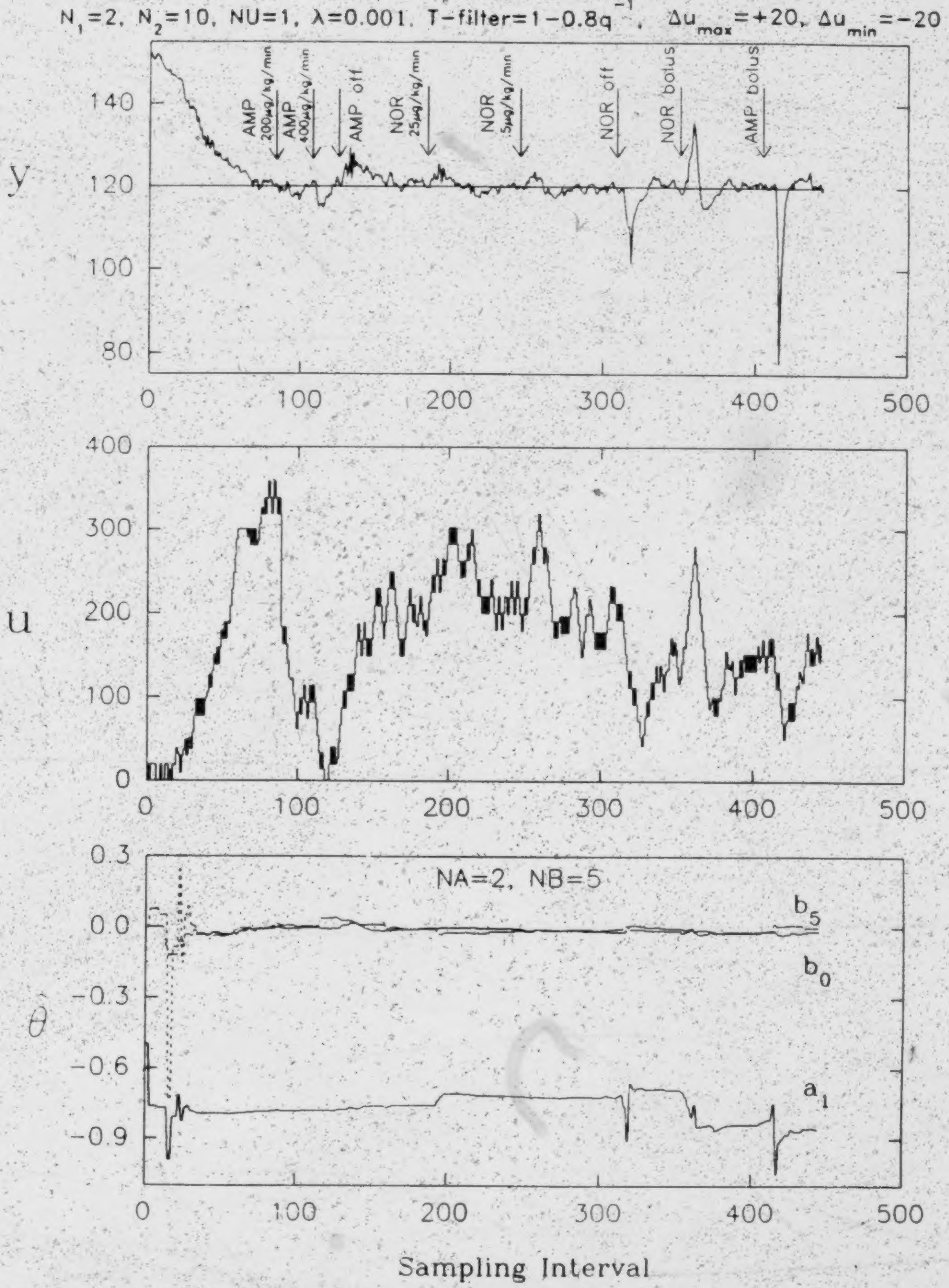


Figure 5.26: Arterial pressure control of a dog with rate constrained GPC

rate constrained GPC is better at handling disturbances.

5.5 Conclusions

The experimental runs conducted on the stirred tank and for controlling the blood-pressure demonstrate that generalized predictive control is a powerful control algorithm capable of effectively controlling SISO and MIMO systems. The algorithm can handle fairly non-linear, open-loop unstable systems. Adaptive GPC can control systems with varying time delays and dynamics. MGPC is capable of handling different time delays in different output channels, and can "optimally" decouple the interaction between various loops, depending on the model and the tuning parameters.

The constrained GPC has distinct advantages over the unconstrained GPC. Rate and amplitude constraints on the manipulated variables can incorporate the physical limitations of the process in the controller. In the experiments, the amplitude constraints of 0% and 100% on minimum and maximum valve openings were incorporated in the control algorithm. This constraints help in avoiding "integral windup". Though the pilot scale stirred tank did not have physical constraints on the rate of change of manipulated variables, it is not uncommon to see them in the industry. The rate and amplitude constraints can be incorporated in the constrained GPC algorithm, to achieve better control of the process. As has been demonstrated by some of the experimental runs, implementation of constraints on the rate of change of manipulated variables can stabilize poorly tuned systems. However, constraints on the manipulated variables cannot change the nature of control action *i.e.* an aggressive controller will remain aggressive with or without the constraints and the ringing action of the control signal will persist independent of constraints. But, rate constraints can limit the rate of change in control signal and thereby improve robustness of the system. Experimental runs also demonstrate that whenever the process output hits the output constraint, than the control algorithm will try and bring back the plant output in feasible region of the constraints.

The tuning of the MGPC for the two input two output system was non-trivial because of the non-linearity of the physical process. The simulation results for linear systems cannot predict the nature of controller settings for a non-linear system. Though the tuning of MGPC was achieved by trial and error, it was much easier than the effort required to tune the PID controllers. Furthermore, the performance of MGPC is far superior to that of PID controllers.

Chapter 6

Scaling in Multivariable Systems

While evaluating the performance of MGPC with the help of simulations (chapter 4) and experimental runs (chapter 5), it was observed that the output scaling factor y_{wt}, is an important tuning knob and it has a significant affect on the performance of the system. This chapter is concerned with the analysis of scaling of controlled and manipulated variables of a process and their affect on the performance of the system.

Scaling of controlled and manipulated variables is often required to yield useful results from various techniques used for designing and analyzing multi-input multi-output systems. The singular value decomposition analysis technique has been widely used in the literature, for the analysis of multivariable control systems (Bonvin and Mellichamp [8], Johnston and Borton [21,22], Keller and Bonvin [24], Lau and Jensen [29], Moore [32,33]). The singular values and the condition number of a matrix can be altered by scaling a matrix by pre-multiplying and/or post-multiplying the matrix with diagonal matrices. Ill-conditioned dynamic matrices often arise due to poor scaling of process variables and computational arithmetic on such ill-conditioned matrices using finite-precision machines can give erroneous results and thus poor control performance. It is possible to improve the performance of a system by scaling the inputs and outputs of a system. Many scaling procedures have been suggested in the literature based on physical arguments (Bonvin and Mellichamp [7,8], Johnston and Borton [22], and minimisation of condition number (Keller and Bonvin [24], Lau and Jensen [29]). However, none of these scaling policies have been evaluated for improvement in numerical stability of ill-conditioned systems on a finite precision computing machine. The effect of scaling of inputs and outputs on the performance of a closed loop feedback system has also not been considered.

This chapter analyzes the effects of scaling on the closed loop performance of a multivariable system. It describes the necessary and sufficient conditions required for a best scaled square matrix in l_2 norm. The effects of scaling are demonstrated on a multi-input multi-output system under long range predictive control (MGPC), with control computation done on a finite precision computing machine. This chapter also differentiates between output and input scaling and gives insight on their individual effects. The optimal scaling method for systems with equal number of inputs and outputs is extended to systems with unequal number of inputs and outputs.

The organization of this chapter is as follows: Section 6.1 describes the motivation for scaling the manipulated and control variables of a process, section 6.2 defines and explains the procedure to scale a matrix optimally, section 6.3 illustrates the effect of scaling in multivariable systems with the help of simulations and explains the individual effects of input and output scaling, section 6.4 extends the scaling procedure to systems with unequal number of inputs and outputs in a process, and section 6.5 concludes the chapter by summing up the cause and effect of scaling the control and manipulated variables.

6.1 Introduction

In multivariable systems, we encounter systems with high gains for some inputs and low gains for others. Such systems arise often due to non-commensurate choice of scaling for the controlled and/or manipulated variables as for example in the simultaneous control of average molecular weight and temperature in a polymer reactor system. This can lead to computational problems in terms of numerical robustness and control. Consider the following two input two output multivariable system:

$$G(s) = \begin{bmatrix} \frac{k_{11}}{s+1} & \frac{k_{12}}{s^2+2s+1} \\ \\ \frac{k_{21}}{s+1} & \frac{k_{22}}{s+1} \end{bmatrix}$$

Let the steady state gain matrix be:

$$K = \begin{bmatrix} k_{11} & k_{12} \\ \\ k_{21} & k_{22} \end{bmatrix} = \begin{bmatrix} 10^{-8} & 1.00 \\ \\ 10^{-6} & 10^{+8} \end{bmatrix} \tag{6.1}$$

The system is discretized with a sampling time of 0.1 unit. The condition number of the steady state gain matrix for the above system is 10^{16}. This system was simulated in a closed loop configuration with the generalized predictive controller, on a computer using MATLAB-386 software, with a numerical accuracy of at least 16 decimal places. The MGPC tuning parameters where set to $N_1 = 1$, $N_2 = 10$, $Q = \mathbf{0}$ and NU = 4. The simulation results for this system are shown in figure 6.1. It is observed that the output y_1 cannot track the set point.

This is due to the high condition number of the steady state gain matrix, which is equal to 10^{16} (in l_2 norm) and the corresponding poor numerical accuracy of the computer. This problem can be easily handled by scaling the inputs and/or outputs of the multivariable systems, as shown in figure 6.2. The matrices D_1 and D_2 are the scaling diagonal matrices, for the process outputs and inputs respectively. The outputs of the system are scaled by the matrix D_1 and appear as scaled inputs to the controller. The controller computes the inputs to the system, assuming that the scaling matrices are a part of the system. The controller outputs are scaled by the matrix D_2 and then implemented on the system.

In this chapter, the effect of scaling multi-input multi-output systems having an ill-conditioned steady state gain matrix is analyzed, in the context of a long range predictive control algorithm–generalized predictive controller (MGPC). The questions addressed pertaining to the scaling are

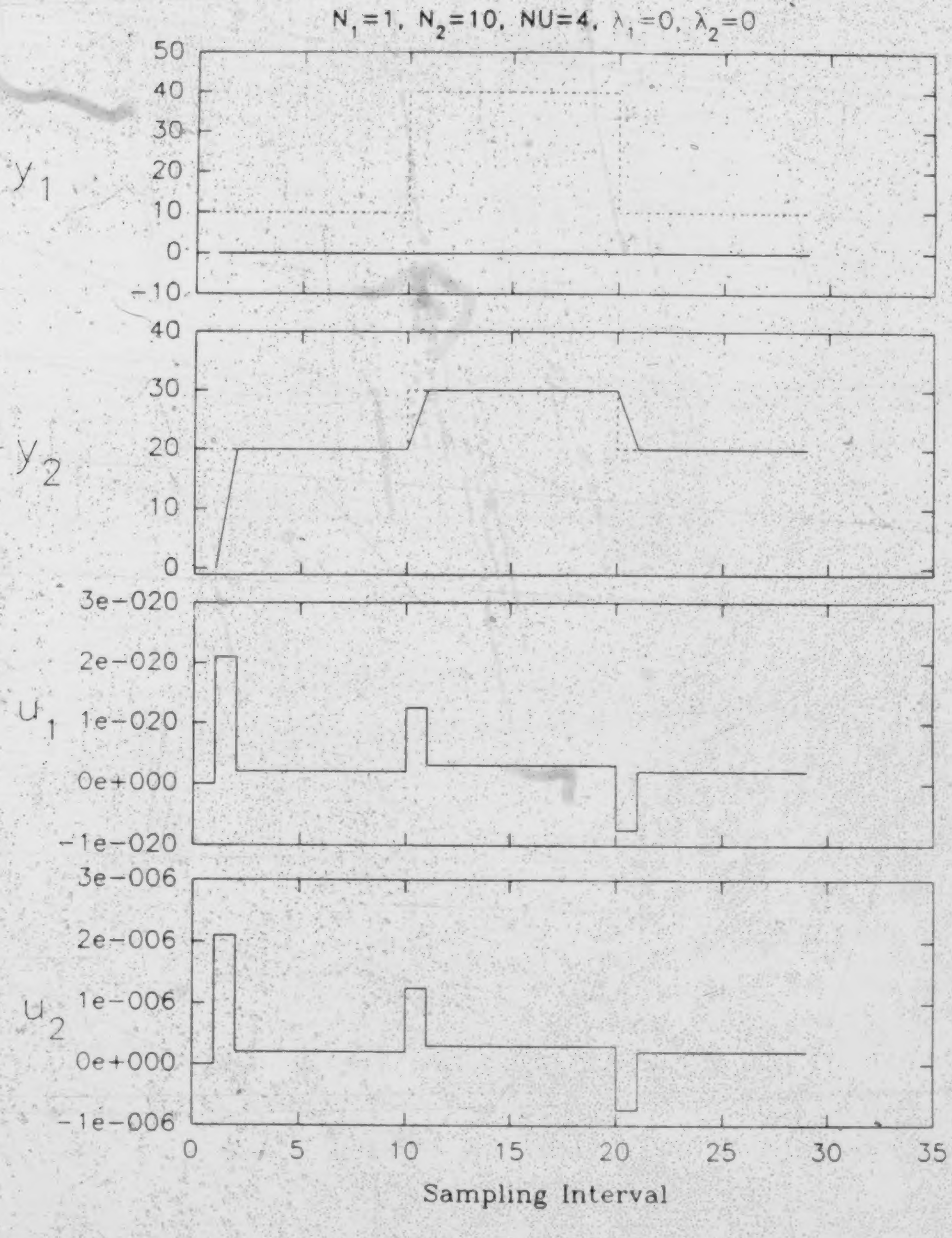


Figure 6.1: Simulation of an ill-conditioned system G(S) with no scaling

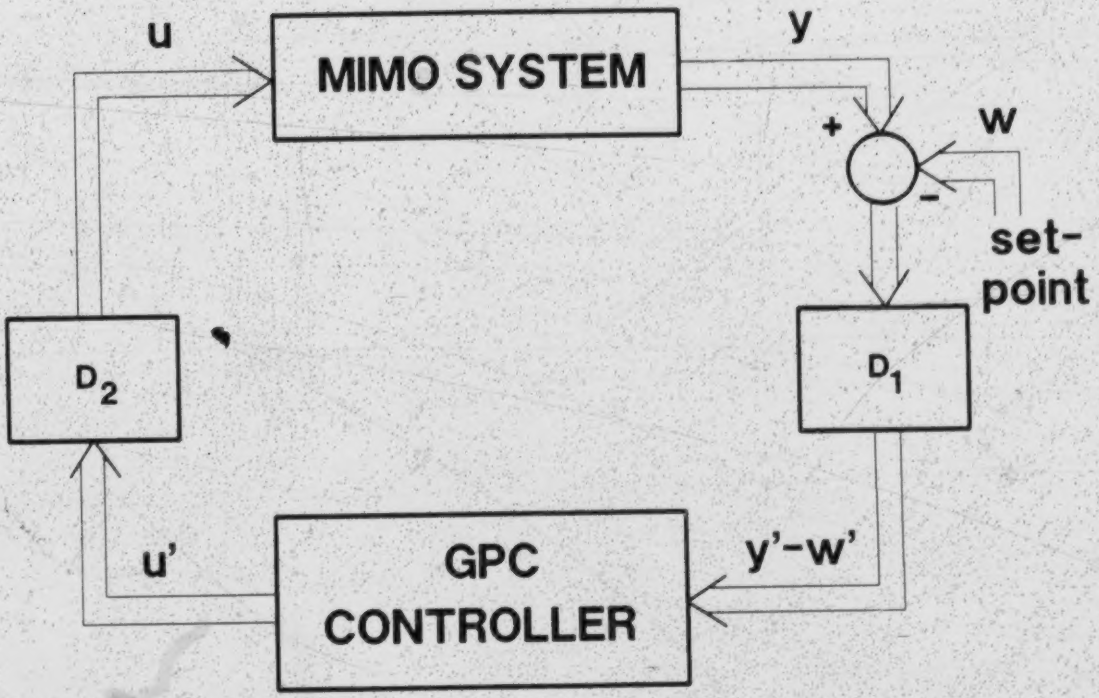

u
MIMO SYSTEM
y
+
w
–
set-
point
D2
D1
u'
GPC
CONTROLLER
y'–w'

Figure 6.2: Block diagram representation of the scaled system

- how do we select the scaling matrices D_1 and/or D_2
- illustration of how scaling matrices D_1 (output scaling) and/or D_2 (input scaling) affect the performance of the overall system, with the help of an example.

6.2 Best l_2 Scaling of a Matrix

Any norm can be used for for evaluating the condition number. The condition number is defined in n-norm as follows

$$\| A \|_n \| A^{-1} \|_n$$

To understand why the l_2 norm (also called the spectral norm) should be used for evaluating the condition number, let us look at the physical interpretation of singular value decomposition (SVD), as described by Moore [33].

6.2.1 SVD—A Physical Interpretation

The singular value decomposition of a matrix K results in three component matrices as follows:

$$K = U \Sigma V^T$$

where

- K is a $n \times m$ matrix (the steady state gain matrix) (usually m $\geq$ n)
- U is a $n \times n$ orthonormal or unitary matrix called the "left singular vector"
- V is a $m \times m$ orthonormal or unitary matrix called the "right singular vector"
- Σ is an $n \times m$ diagonal matrix of scalers called the "singular values", which are organized such that $\sigma_1 > \sigma_2 \cdots > \sigma_n > 0$

The singular values σ_i provide the ideal decoupled gain of the i^{th} open loop process and are a direct measure of the decoupled gains of the process. If the multivariable interactions were removed by an ideal "steady-state" decoupler, the singular values would be the open loop gains of the noninteracting system.

Consider $\mu = \sigma_{max}/\sigma_{min} = \sigma_1/\sigma_n$. μ is the condition number of the matrix K in the l_2 norm. A large μ indicates that the relative sensitivity of the multivariable system is very uneven. Also, the larger the μ the more difficult it is to compute the inverse of the matrix that is free of computational errors. All predictive controllers inherently do inversion of some function of the transfer function matrix of the system, to calculate the control signals which minimize the error between the outputs and the setpoints of the system. If K is the steady state model of the system, then

$$K^{-1} = V\Sigma^{-1}U^T$$

Therefore, accuracy in inversion of any matrix is equivalent to accuracy in inversion of its singular value matrix Σ. Inversion of matrix Σ is equal to the transpose of the matrix with non-zero elements of Σ replaced by their reciprocals. In l_2 norm the condition number of matrices K and Σ are equal to $\mu = \sigma_{max}/\sigma_{min}$ (since $\| A \|_2 = \sigma_{max}$ and $\| A^{-1} \|_2 = \sigma_{min}$). This is equal to the ratio of the strongest and the weakest steady state gains of the ideally decoupled multivariable system. This is a good measure of the condition of the multivariable system because it is the ratio of maximum and minimum energy transfer from inputs to outputs, for the ideally decoupled system. This should be close to unity for good or even "controllability" of all the outputs. Because of this physical significance, the use of l_2-norm for calculating the condition number of the system is more appropriate.

6.2.2 Condition for Best Scaling of a Matrix

Let K be the $n \times m$ steady state gain matrix. The question to pose is: what are the best output and input scaling diagonal matrices (D_1 and D_2) of K in the l_2 norm such that the condition number of $D_1 K D_2$ is a minimum, *i.e.*

$$\min_{D_1, D_2} (\sigma_1(D_1 K D_2)/\sigma_n(D_1 K D_2))$$

where $\sigma_1(D_1 K D_2) \geq \sigma_2(D_1 K D_2) \geq \cdots \sigma_n(D_1 K D_2) > 0$ are the singular values of $D_1 K D_2$.

The problem of matrix scaling has been discussed by Lau and Jensen [29], where the authors describe various methods of scaling a steady state gain matrix, to minimize the condition number in l_2 norm. Golub and Varah [20] describe the necessary and sufficient condition for best two sided scaling of a matrix, in l_2 norm. This property is referred to as EMC (*equal modulus components* of the first and last columns of the left and right singular vectors). Any matrix which satisfies the EMC property has the minimum condition number in the l_2 norm.

6.2.2.1 EMC Property

Let $A = U \Sigma V^T$ be the singular value decomposition of a square matrix A, of order $n \times n$. Let u_{ij} and v_{ij} denote the ith element of the jth column of U and V respectively. Then A is best scaled in the l_2 norm if $| u_{i1} | = | u_{in} |$, $| v_{i1} | = | v_{in} |$ for $i = 1, \ldots,$ n. That is, A is best scaled if the first and last columns of U and V have components of equal magnitude.

Golub and Varah [20] show that if a square matrix A, with simple (distinct) σ_1 and σ_n, is in the best scaled form, then the EMC property is **necessary and sufficient** for best two sided l_2 scaling. For multiple σ_1 and σ_n, the EMC property is only **sufficient** for optimal two sided scaling.

For one sided scaling, only one of U or V is involved in the EMC property. For optimal scaling (D_1) of the outputs, U needs to satisfy the EMC property and for optimal scaling (D_2) of the inputs, V needs to satisfy the EMC condition, assuming σ_1 and σ_n are distinct.

By using the EMC condition, it is possible to verify if a given matrix is optimally scaled in l_2. Function minimizing procedures can be used for finding the diagonal matrices, which scale the given matrix optimally, to minimize the condition number. If the set of matrices satisfy the EMC condition, then these diagonal matrices are the best choices for scaling. Since only the relative values of the elements in a scaling matrix are important, one of the element in any diagonal scaling matrix can be chosen arbitrarily. In the examples discussed below, a simplex algorithm available in MATLAB (NELDER or FMINS) was used for finding the optimal scaling diagonal matrices such that the EMC property is satisfied.

Example: Consider a matrix:

$$A = \begin{bmatrix} 1.0 & 2.0 & 3.0 \\ 1.0 & -1.0 & 1.0 \\ 0.0 & 1.0 & 1.0 \end{bmatrix}$$

The singular value decomposition of $A = U_1\Sigma_1V_1$ results in:

$$U_1 = \begin{bmatrix} 0.9327 & 0.0493 & 0.3572 \\ 0.1476 & -0.9603 & -0.2399 \\ 0.3312 & 0.2747 & -0.9027 \end{bmatrix} \quad V_1 = \begin{bmatrix} 0.2682 & -0.5352 & 0.8010 \\ 0.5122 & 0.7834 & 0.3520 \\ 0.8159 & -0.3159 & -0.4842 \end{bmatrix}$$

$$\Sigma_1 = \mathrm{diag}(4.0101,\ 1.7022,\ 0.1465)$$

and the condition number of A is approximately 27.4. The optimal scaling matrices D_1 and D_2 are:

$$D_1 = \begin{bmatrix} 1.0000 & 0.0000 & 0.0000 \\ 0.0000 & 1.7321 & 0.0000 \\ 0.0000 & 0.0000 & 3.0000 \end{bmatrix} \quad D_2 = \begin{bmatrix} 1.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.5000 & 0.0000 \\ 0.0000 & 0.0000 & 0.4082 \end{bmatrix}$$

The singular value decomposition of $D_1AD_2 = U_2\Sigma_2V_2$ gives:

$$U_2 = \begin{bmatrix} 0.7071 & 0.0000 & 0.7071 \\ 0.3536 & -0.8660 & -0.3536 \\ 0.6124 & 0.5000 & -0.6124 \end{bmatrix} \quad V_2 = \begin{bmatrix} 0.5000 & -0.7071 & 0.5000 \\ 0.5000 & 0.7071 & 0.5000 \\ 0.7071 & 0.0000 & -0.7071 \end{bmatrix}$$

$$\Sigma_2 = \mathrm{diag}(2.6388,\ \ 2.1283,\ \ 0.1895)$$

The condition number of D_1AD_2 is approximately 13.9. Note that for the matrices U_2 and V_2 the magnitudes of the elements in first and last column are equal. Hence, the matrices D_1 and D_2 scale the matrix A optimally, as they satisfy the EMC condition.

However, it should be noted that the best scaled matrices may still have high condition number. For instance, a matrix with linear dependent rows or columns has a infinite condition number and if optimally scaled it may still have a large condition number. This is further illustrated by the following example:

Example: Consider an example from Skogestad *et al.* [51]. The steady state gain matrix of the two input two output system is

$$A = \begin{bmatrix} 0.878 & -0.864 \\ 1.082 & -1.096 \end{bmatrix}$$

The singular value decomposition of $A = U_1\Sigma_1V_1$ gives:

$$U_1 = \begin{bmatrix} 0.625 & 0.781 \\ 0.781 & -0.625 \end{bmatrix} \quad V_1 = \begin{bmatrix} 0.707 & 0.708 \\ -0.708 & 0.707 \end{bmatrix}$$

$$\Sigma_1 = \mathrm{diag}(1.972,\ \ 0.0139)$$

and the condition number of A is approximately 141.7. The optimal scaling matrices D_1 and D_2 are $D_1 = \mathrm{diag}(1.000,\ 0.7998)$ $D_2 = \mathrm{diag}(1.000,\ 1.0016)$

The singular value decomposition of $D_1AD_2 = U_2\Sigma_2V_2$ gives:

$$U_2 = \begin{bmatrix} 0.7071 & -0.7071 \\ 0.7071 & 0.7071 \end{bmatrix} \quad V_2 = \begin{bmatrix} 0.7071 & -0.7071 \\ -0.7071 & -0.7071 \end{bmatrix}$$

$$\Sigma_2 = \mathrm{diag}(1.7434,\ 0.0126)$$

The condition number of D_1AD_2 is approximately 138.2. The matrices D_1 and D_2 scale the matrix A optimally, as they satisfy the EMC condition. However, the condition number of the optimally scaled matrix D_1AD_2 could not be decreased substantially. This is because of the approximate linear dependency of the rows (or columns) in matrix A. This results in both the inputs u_1 and u_2 affecting the outputs y_1 and y_2 similarly *i.e.* the system is weakly output controllable. This is an inherent design problem. No amount of scaling can alter the fundamental controllability and observability of the process. Such problems cannot be corrected by scaling the inputs and/or outputs.

6.3 Use of Scaling In MIMO Systens

It is a well know fact that ill-conditioned systems cause control problems. If the system is ill-conditioned due to linear dependency of rows or columns in the steady state gain matrix of the system, it cannot be controlled easily, as illustrated in the previous section. Such problems can only be corrected by physical plant re-design or modifications. But if the system is ill-conditioned due to some of the system gains being large and thus strongly dependent on some of the inputs and weakly dependent on others, then scaling can help. In this chapter, the effect of scaling on multi-input multi-output systems with an ill-conditioned steady state gain matrix is analyzed, in a closed-loop situation with the controller based on a long range predictive control algorithm–generalized predictive controller of Clarke *et al.* [13,14]. The multivariable extension of the algorithm is described in Mohtadi [35] and Shah *et al.* [46]. The system is scaled by diagonal scaling matrices as shown in figure 6.2. The outputs of the process are scaled through the matrix D_1. The scaled outputs are used by the controller to compute the inputs to the system, assuming that the scaling matrices are a part of the system. The controller outputs are scaled through the matrix D_2 and are implemented as inputs to the system. The elements of the scaling matrices can have either positive or negative signs. In this chapter, all the elements of scaling matrices were used with positive sign.

6.3.1 Formulation of Scaled MGPC

This section presents the mathematical formulation of a manipulated and control variables scaled MGPC. Consider the following multi-input multi-output description of a linear deterministric process:

$$\tilde{\mathcal{A}}(q^{-1})\mathbf{y}(t) = \mathcal{B}\Delta\mathbf{u}(t-k) \tag{6.2}$$

where

- $\mathbf{y}(t)$ and $\Delta\mathbf{u}(t)$ are m x 1 output and incremental input vectors
- $\mathcal{A}(q^{-1})$ and $\mathcal{B}(q^{-1})$ are polynomial matrices in the backward shift operator and
- $\tilde{\mathcal{A}}(q^{-1}) = \mathcal{A}(q^{-1})\Delta$, where $\Delta = 1 - q^{-1}$.

Recall the cost function used by MGPC (equation 2.19) to calculate the control signal:

$$\begin{aligned} J_{MGPC} &= \sum_{j=N_1}^{N_2} [\hat{\mathbf{y}}(t+j) - \mathbf{w}(t+j)]^T[\hat{\mathbf{y}}(t+j) - \mathbf{w}(t+j)] \\ &\quad + \sum_{j=1}^{NU} \Delta\mathbf{u}(t+j-1)^T \Lambda \Delta\mathbf{u}(t+j-1) \end{aligned} \tag{6.3}$$

where

- $\hat{\mathbf{y}}$ is the system output
- $\Delta\mathbf{u}$ is controller output
- N_1 and N_2 are lower and upper prediction horizons
- NU is the control horizon
- $\Lambda(j)$ is the control weighing matrix, and
- $\mathbf{w}(t+j)$ is the desired output or set-point.

The MGPC controller assumes that the scaling matrices D_1 and D_2 are a part of the system. If $\Lambda(j)$ is a non-zero matrix in any specific problem for the unscaled system, then $\Lambda(j)$ may have to be modified, after scaling, as per the performance requirements. The $\sum_{j=0}^{NU} \Delta\mathbf{u}(t+j)^T \Lambda(j) \Delta\mathbf{u}(t+j)$ term in equation 6.3 is omitted for brevity, as scaling is independent of it. The equation representing the system with scaling matrices is:

$$\tilde{\mathcal{A}}_s(q^{-1})\mathbf{y}'(t) = \mathcal{B}_s(q^{-1})\Delta\mathbf{u}'(t-1) \tag{6.4}$$

where $(\cdot)_s$ denotes the scaled matrix polynomial and the relation between the unscaled and scaled system is given by:

$$\tilde{\mathcal{A}}_s = \tilde{\mathcal{A}} {D_1}^{-1} \tag{6.5}$$

$$\mathcal{B}_s = \mathcal{B} D_2 \tag{6.6}$$

The cost function of MGPC for this scaled system is:

$$\min J_{scaled} = \sum_{j=N_1}^{N_2} [\hat{\mathbf{y}}'(t+j) - \mathbf{w}'(t+j)]^T [\hat{\mathbf{y}}'(t+j) - \mathbf{w}'(t+j)] \tag{6.7}$$

This system is related to the unscaled system, as can be seen from figure 6.2 by the following equations:

$$\hat{\mathbf{y}}'(t+j) = D_1 \hat{\mathbf{y}}(t+j) \tag{6.8}$$

$$\mathbf{w}'(t+j) = D_1 \mathbf{w}(t+j) \tag{6.9}$$

$$\Delta\mathbf{u}(t+j) = D_2 \Delta\mathbf{u}'(t+j) \tag{6.10}$$

With this mathematical formulation of the scaled MGPC, the next subsection illustrates the various possible ways of scaling and their effects. A simple example shows the effects of various types of scaling.

6.3.2 Scaling And Their Effects

6.3.2.1 Best D_1 Scaling Of The Steady State Gain Matrix In MGPC

The best D_1 scaling matrix is found by using a search procedure on the steady state gain matrix K, such that the condition number of D_1K is minimum. D_2 is set equal to the identity matrix. Matrix U from singular value decomposition of scaled steady state gain matrix $D_1K = U\Sigma V$, is verified for satisfying the EMC condition.

The scaling matrix D_1 scales the system outputs (figure 6.2). This scaling can be interpreted as being equivalent to output weighting. The objective function of scaled system (equation 6.7) gives variable weight to the different outputs of the system. The equation 6.7 can be expanded as:

$$J_{scaled} = \sum_{j=N_1}^{N_2} [D_1(\hat{\mathbf{y}}(t+j) - \mathbf{w}(t+j))]^T [D_1(\hat{\mathbf{y}}(t+j) - \mathbf{w}(t+j))] \qquad (6.11)$$

if D_1 and $e(t+j)$ are defined as:

$$\begin{aligned} D_1 &= \mathrm{diag}(d_1, d_2, \cdots, d_n) \\ \mathbf{e}(t+j) &= [e_1\ e_2\ \cdots\ e_n]^T = \hat{\mathbf{y}}(t+j) - \mathbf{w}(t+j) \end{aligned}$$

then equation 6.11 can be rewritten as:

$$J_{scaled} = \sum_{j=N_1}^{N_2} [d_1e_1(t+j)\ \ d_2e_2(t+j)\ \cdots\ d_ne_n(t+j)][d_1e_1(t+j)\ \ d_2e_2(t+j)\ \cdots\ d_ne_n(t+j)]^T$$

Which means that the control objective is being changed, such that the weight given to different outputs is proportional to the element of the diagonal scaling matrix D_1.

6.3.2.2 Best D_2 Scaling of the Steady State Gain Matrix in MGPC

The best D_2 diagonal scaling matrix is found by using a search procedure on the steady state gain matrix K, so as to minimize the condition number of KD_2, assuming D_1 to be an identity matrix. Then the right singular vector matrix V of KD_2 is verified for satisfying the EMC condition.

Unlike D_1, which scales the system outputs, D_2 does not affect the control law and it gives better numerical conditioning for the system. This helps in improving the performance of ill-conditioned systems. The controller has the model of the process including matrix D_2 as a part of the model. Therefore, the control law computes the inputs to the system : GD_2.

The condition number of the scaled steady state gain matrix KD_2 is much lower than the unscaled system. The inversion of matrix KD_2 can be performed with better numerical accuracy. The controller inverts a matrix related to KD_2. Hence,the control law is numerically more robust for properly scaled system. The equivalence of the control law for unscaled and the D_2 scaled system is shown in appendix B. The scaling matrix D_2 increases the numerical robustness of the overall system, without affecting the control law.

6.3.2.3 Best Two Sided Scaling Of the Steady State Gain Matrix in MGPC

The matrices D_1 and D_2 for scaling are found by a search procedure and verified for satisfying the EMC condition. The system is scaled on both sides by the matrices D_1 and D_2. This gives the best condition number, among the three possible choices. It increases the numerical stability of the process and also changes the control law, by giving variable weight to the system outputs, as explained above.

6.3.3 Simulation Example

To illustrate the effect of the various methods of scaling, the same example simulated in section (6.1) of the unscaled system is used. All the parameters of the controller are set as in the unscaled case.

6.3.3.1 Optimal Output Scaling

The system is optimally scaled with D_1, which is found by a search procedure and verified for satisfying the EMC condition. D_2 is set to the identity matrix.

$$D_1 = \begin{bmatrix} 1.00 & 0.00 \\ 0.00 & 10^{-8} \end{bmatrix}$$

The condition number of the matrix $D_1 K$ is 2 x 10^8. The results from the simulation of this system are shown in figure 6.3. In this simulation the set point tracking of y_1 is achieved. But the performance of y_2 is not acceptable.

6.3.3.2 Optimal Input Scaling

The system is optimally scaled with D_2. D_1 is set to identity matrix.

$$D_2 = \begin{bmatrix} 1.00 & 0.00 \\ 0.00 & 10^{-14} \end{bmatrix}$$

The condition number of the matrix $K D_2$ is 200.01. The simulation of this system is shown in figure 6.4. The performance of this system is satisfactory.

6.3.3.3 Optimal Input and Output Scaling

The system is optimally scaled with D_1 and D_2, which are found by a search procedure and verified for satisfying the EMC condition.

$$D_1 = \begin{bmatrix} 1.00 & 0.00 \\ 0.00 & 10^{-5} \end{bmatrix} \quad D_2 = \begin{bmatrix} 1.00 & 0.00 \\ 0.00 & 10^{-11} \end{bmatrix}$$

The condition number of the system $D_1 K D_2$ is 1.002. The simulation of this system is shown in figure 6.5. This setup has the minimum possible condition number. But the set point tracking of y_2 is still not acceptable.

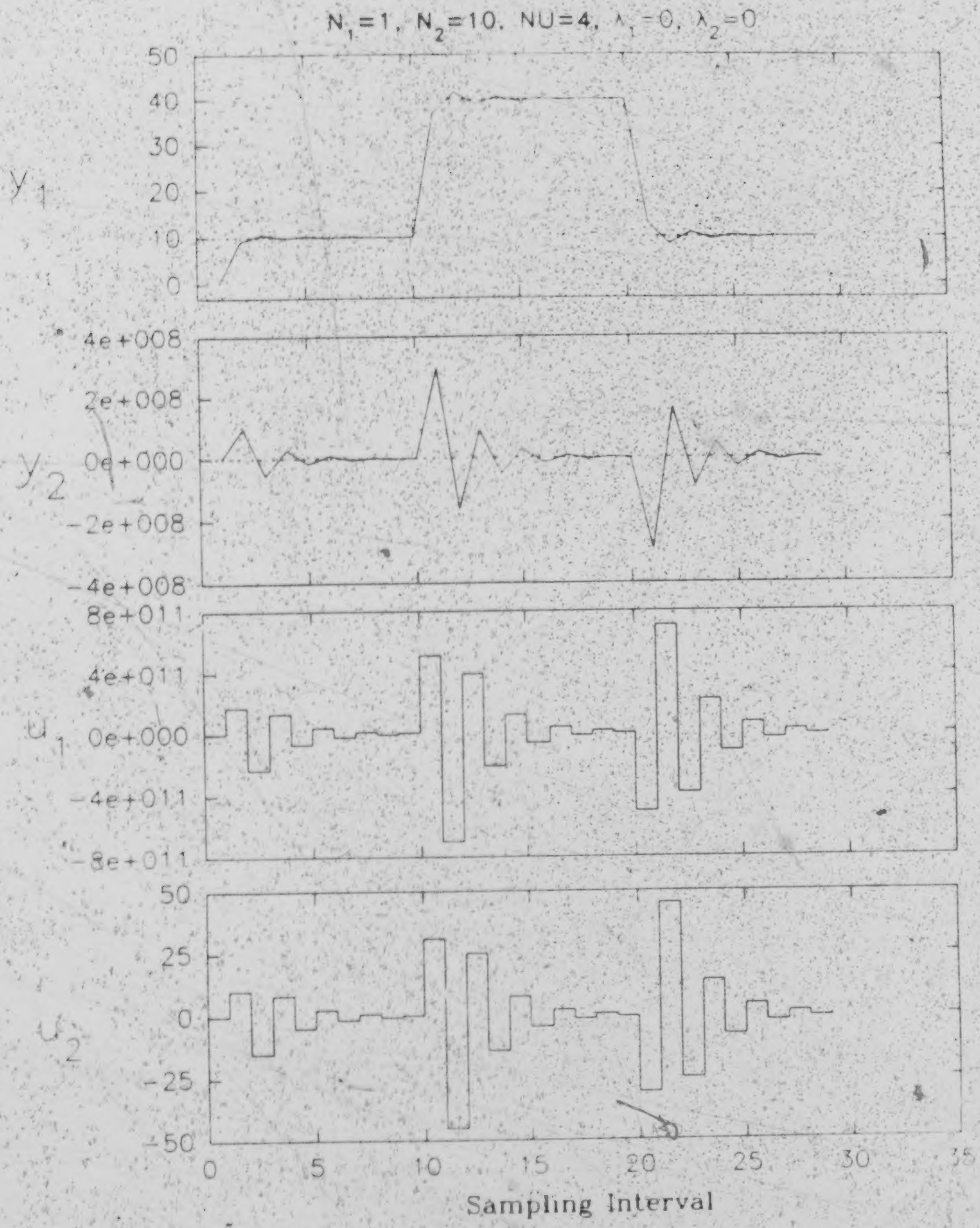


Figure 6.3: Simulation of an ill-conditioned system with system outputs scaled

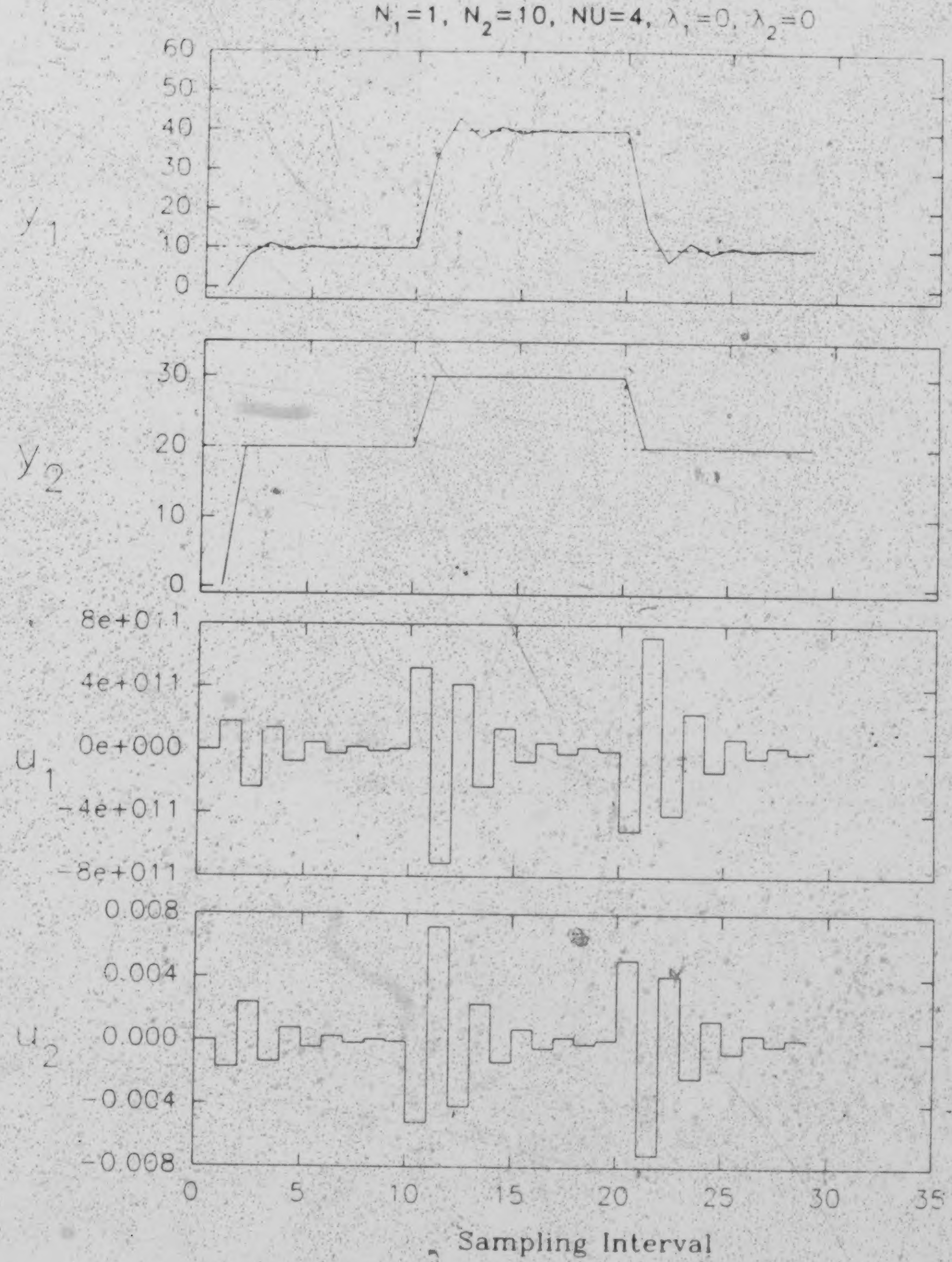


Figure 6.4: Simulation of an ill-conditioned system with system inputs scaled

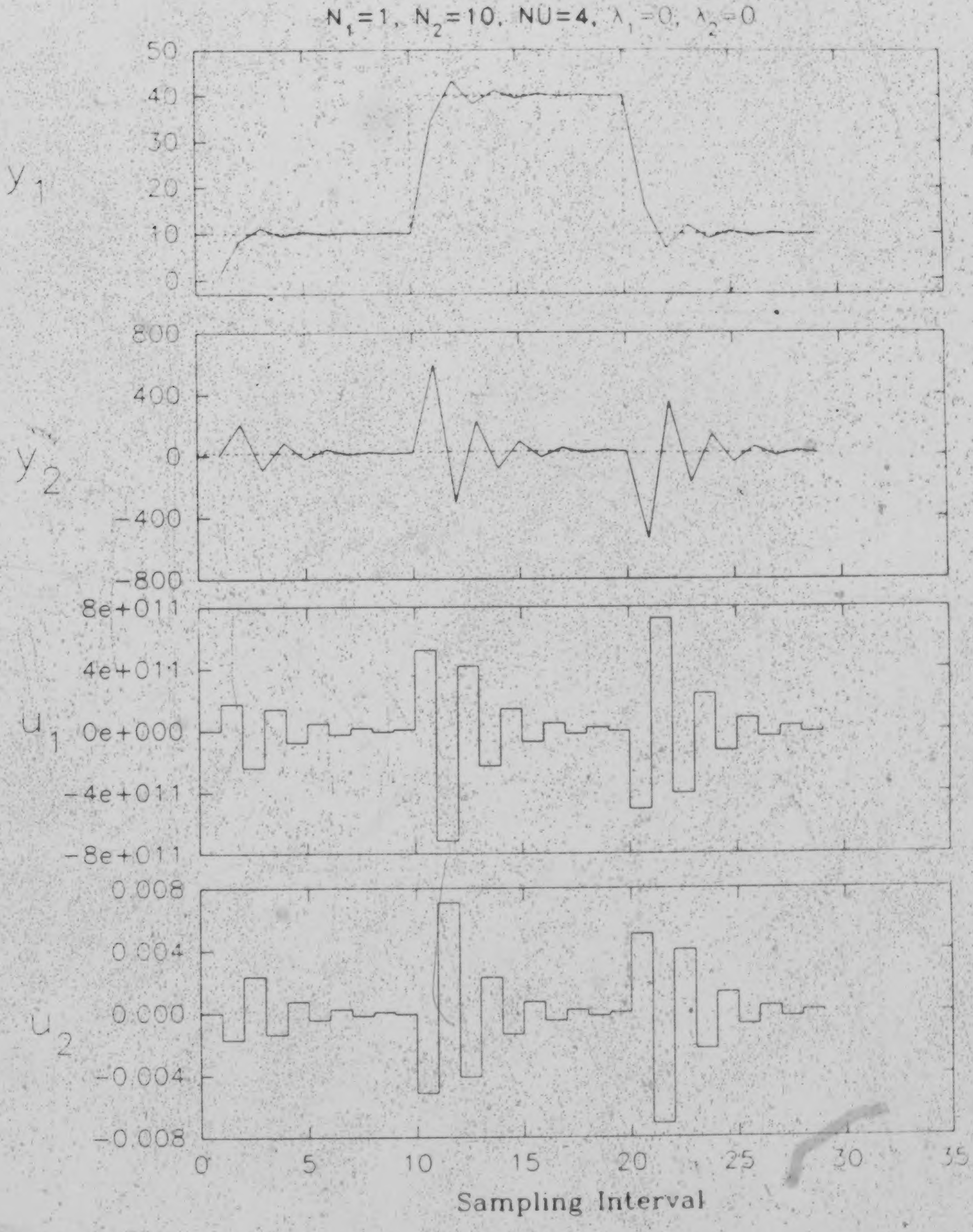


Figure 6.5: Simulation of an ill-conditioned system with both system inputs and outputs scaled

6.3.4 Discussion

Output only Scaling with D_1: From the simulation results, it is observed that the outputs scaling matrix D_1 and inputs scaling matrix D_2 affect the performance of the control in different ways. If the outputs are scaled with a non-identity matrix D_1, then the objective function of MGPC gets modified. In this example, the objective function gives relatively high weight to the control of output y_1 than to y_2 by a factor of 10^8. Hence, the modified objective function ignores the error in the second output and does not track the set point changes properly. As is evident from the simulation results, this may not lead to the best possible control.

In the discrete linear quadratic regulator (LQR) problem, the objective is to find control vectors $u(t), u(t+1), \ldots$ that minimizes the performance index of the expression of the form

$$J = \sum_{j=0}^{N-1} x(t+j)^T Q x(t+j) + u(t+j)^T R u(t+j) \tag{6.12}$$

If we assume that the states of the system are the same as the outputs of the multivariable system, then the perturbation variable of states is

$$x(t+j) = y(t+j) - w(t+j)$$

where y is the output vector and w is the setpoint vector. The performance index of equation 6.12 is similar to objective index of output scaled MGPC in equation 6.11. The matrix Q in equation 6.12 is equal to $D_1^T D_1$ in equation 6.11. Some recommended procedure available in the literature for the selection of Q matrix in LQR problem can also be used for selecting the D_1 matrix in equation 6.11. It has been suggested in the literature that the state weighting matrix Q in the LQR problem can be selected from physical arguments (Åström and Wittenmark [4]). It is also pointed out that a particular difficulty exists in finding the relative weights between the state variables, which can be done by trial and error. Sometimes the specification of the system is given in terms of the maximum allowed disturbance in states. One rule of thumb to decide the weighting in equation 6.12 is to choose the diagonal elements as the inverse value of the square of the allowed deviations. This and many other similar arguments in selecting the matrix Q in LQR problem can be used in selecting the D_1 matrix for scaling the outputs of a multivariable system.

The simulations of the Shell control problem in section 4.2 and the experimental runs with the stirred tank heater in section 5.3.3.4 show that large output weighting on outputs with large model process mismatch enhance the performance of the system. This criterion of selecting the scaling matrix D_1 will result in a better performance of the system.

Input only Scaling with D_2: By scaling the inputs with the matrix D_2 based on the EMC procedure, the objective function is not modified. This scaling gives numerical robustness to the controller and generates better control signals. The condition number of the steady state gain matrix of the scaled system (KD_2) is much lower than the condition number of the unscaled steady state gain matrix. Hence, the numerical stability of the control algorithm with the scaling matrix D_2 is better. D_2 acts as a multiplication factor without modifying the control law. This scaling has no negative side effects. It is shown in the appendix B that the MGPC control law for the scaled and the unscaled system is invariant of D_2 for an infinite precision machine. D_2 affects numerical computation in the MGPC control calculation since D_2 is a pre-multiplier to the process.

Both Output and Input Scaling with D_1 and D_2: When the system is scaled at both outputs and inputs, the least condition number for the system (1.002) is achieved. But it does not produce the best results. Though, numerical stability is created in the system by the use of the scaling matrix D_2, the controllers objective function is modified by the matrix D_1. The net result is that large weights are given to control some outputs than others, *e.g.* in the above example y_1 is given relatively large weight in comparison to y_2.

6.4 Scaling for Systems with an Unequal Number of Inputs and Outputs

Consider a system of n outputs and m inputs (with $m > n$). There are C_n^m submatrix combinations of order n x n of steady state gain matrix K of the system. Golub and Varah [20] have given a conjecture for best scaling of a rectangular matrix. They state that there exists an n x n submatrix of K which, when best scaled, gives the best scaling for K also. This can be one of the many best scaling matrices and is not necessarily unique. It is not easy to scale all the submatrices of K and then pick the best scaled matrix among them. A suboptimal alternative while scaling can be achieved as follows:

- Select a square submatrix of K such that the submatrix has least condition number among the C_n^m submatrices. The submatrix can be chosen by the procedure described by Keller and Bonvin [24], for selection of dominant inputs. This procedure is further elaborated in the next subsection.

- Scale the submatrix with the help of the EMC property and find diagonal scaling matrices D_1 and D_2.

- Let min-element = smallest diagonal element of D_2 or

  = smallest diagonal element of $D_2/10.0$.

- For the columns not considered in submatrix, a value equal to min-element is used in the scaling matrix. The condition number of the submatrix and final scaled matrix are almost equal.

This procedure is intuitively appealing because we are selecting a submatrix with minimum condition number among the various submatrices to ensure that the inputs selected in the submatrix have better controllability of the system among the other possible input combinations. The columns not present in the submatrix are equivalent to being scaled with a zero. After finding the optimal scaling for the submatrix, we increase the weight of scaling of the columns from zero weight to a value less than or equal to the smallest scaling factor. This ensures that the condition number of the scaled rectangular steady state gain matrix is almost equal to the condition number of the scaled submatrix.

6.4.1 Selection of the Submatrix

In the above procedure, the selection of submatrix K_{sub} from the steady state gain matrix K (n x m matrix, $m > n$) should be done as described below.

The singular value decomposition of the steady state gain matrix K can be represented as

$$K = U\Sigma V^T = U \begin{bmatrix} \Sigma_1 & \vdots & 0 \\ 0 & \vdots & 0 \end{bmatrix} \begin{bmatrix} V_1^T \\ \dots \\ V_2^T \end{bmatrix}$$

where Σ_1 contains the singular values and V_1 has the corresponding right singular vectors.

Consider the symmetric matrix

$$P = V_1 V_1^T$$

of dimension m x m.

The diagonal element P_{ii} is the fraction of the mode excitation stemming from the input u_i that is transmitted through Σ_1. The following procedure, as described by Keller and Bonvin [24], can be used to select the n dominant inputs, for selecting the submatrix K_{sub}:

- Locate the rows of P for which the diagonal element is nearly one and the off-diagonal elements are small (close to zero). The inputs corresponding to the rows located need to be retained.
- Eliminate the rows (and the columns) with small diagonal elements.

- Make sure that for each column of V_1 there is at least one relatively large element corresponding to a retained input.

The submatrix K_{sub} is formed of inputs selected from the above procedure. This submatrix has least condition number among the possible C_n^m other possible submatrices of dimension n x n. The condition number of the steady state gain matrix and the submatrix are usually of the same order. This square submatrix is then scaled with the help of EMC property.

6.4.2 Example

The above procedure has been applied to an example with four outputs and six inputs, from Bonvin and Mellichamp [8]. The unscaled steady state gain matrix of the system is

$$K = \begin{bmatrix} 2.434e-1 & -1.180e-2 & 1.122e-1 & -1.475e-3 & -2.001e-20 & -2.280e-19 \\ 2.496e+2 & -1.211e+1 & 4.715e+1 & -1.513e+0 & -1.108e-01 & 4.660e+02 \\ -2.697e+0 & 1.151e+0 & 1.973e+1 & 1.439e-1 & 4.248e-18 & 4.840e-17 \\ -2.791e+0 & 1.986e+0 & 3.660e+1 & 2.482e-1 & 9.044e-02 & 1.030e+00 \end{bmatrix}$$

The condition number of the matrix K is 3.9240e+04. A submatrix with inputs 1, 3, 5 and 6 was selected, using the procedure described by Keller *et al.* [24], which had the smallest condition number among the various other possible submatrices of K. The K_{sub} had a condition number of 3.92299e+04.

$$K_{sub} = \begin{bmatrix} 2.434e-01 & 1.122e-01 & -2.001e-20 & -2.280e-19 \\ 2.496e+02 & 4.715e+01 & -1.108e-01 & 4.660e+02 \\ -2.697e+00 & 1.973e+01 & 4.248e-18 & 4.840e-17 \\ -2.791e+00 & 3.660e+01 & 9.044e-02 & 1.030e+00 \end{bmatrix}$$

The diagonal scaling matrices D_1 and D_{2sub} are calculated with the help of EMC condition to minimize the condition number of $D_1 K_{sub} D_{2sub}$.

$$D_1 = \text{diag}(\ 1.0e+0,\ 6.518e\text{-}5,\ 2.172e\text{-}2,\ 1.019e\text{-}3)$$

$$D_{2sub} = \text{diag}(\ 1.0e+0,\ 5.769e\text{-}1,\ 2.757e+3,\ 8.403e+0\ )$$

The condition number of the scaled system $D_1 K_{sub} D_{2sub}$ is 1.101118. To extend the scaling of the sub-system to the complete system, define:

min-element= smallest non-zero element of diagonal matrix D_{2sub} = 5.769e-01.

For the columns 2 and 4 of matrix K which were not scaled while scaling the sub-matrix K_{sub}, an element less than or equal to min-element is used in the scaling matrix D_2

$$D_2 = \text{diag}(\ 1.0e+0,\ 5.769e\text{-}1,\ 5.769e\text{-}1,\ 5.769e\text{-}1,\ 2.757e+3,\ 8.403e+0\ )$$

The condition number of the matrix $D_1 K D_2$ is 1.1015. This procedure gives a good scaling for systems with unequal number of inputs and outputs. This procedure can also be used for one sided scaling of multivariable systems. For only inputs being scaled *i.e.* D_1 set as identity matrix and scaling with D_2 , the condition number of the system discussed above was reduced from 3.924e+04 to 1.995e+03 using the scaling matrix

$$D_2 = \text{diag}(\ 1.0e+0,\ 5.447e\text{-}1,\ 8.500e\text{-}1,\ 5.447e\text{-}1,\ 2.024e+2,\ 5.447e\text{-}1\ ).$$

6.5 Conclusions

Numerical robustness can be incorporated in a given system by scaling the inputs with matrix D_2, so as to minimize the condition number. The outputs scaling matrix D_1 should **not** be decided on the basis of condition number. It must be based on control and stability criterions. The choice of an output scaling matrix is similar to the choice of the state weighting matrix in the classical optimal control problem. The guidelines available in the literature for the selection of state weighting matrix in the LQR problem can also be used for the selection of output scaling matrix D_1. The simulations of the Shell control problem (section 4.2.4.4) and experimental runs conducted on a stirred tank heater (section 5.3.3.4) show that larger output weighting on outputs with larger model process mismatch will result in a better overall performance of the closed loop system.

Numerical robustness can be achieved by scaling inputs with the matrix D_2, so as to minimize the condition number of KD_2 (where K is steady state gain matrix). Input scaling does not modify the cost function of the controller. The matrix D_2 should be selected such that KD_2 fulfills the EMC condition.

For designing the scaling matrices, choose a output scaling matrix based on the physical process (model process mismatch in different outputs, allowed deviations for different output specifications etc). The input scaling matrix must be selected such that D_1KD_2 satisfy the EMC condition.

Also, the EMC condition can be used in other control applications to minimize condition numbers. In the improved least square identification algorithm of Rao [53], the scaling matrix has been calculated using l_∞ norm, since this is computationally easier for on-line calculations. The use of l_2 norm would be more appropriate (see Niederlinski [39]) for calculating the scaling matrices if computation power is not a limitation.

Chapter 7

Conclusions

This thesis has evaluated the performance of a constrained long range predictive control algorithm, the generalized predictive control (GPC) for single-input single-output (SISO) and multi-input multi-output (MIMO) processes. The performance of adaptive constrained GPC was evaluated on SISO processes and the performance of non-adaptive (or fixed parameter) GPC was evaluated on MIMO processes. Constraints were imposed on both the amplitude and the rate of change of control signals and also on the amplitude of process outputs. The thesis has also focused on the analysis of the effects of scaling the inputs and outputs in a multivariable system with some guidelines for scaling them.

The main contributions of the thesis are presented below in a logical order to indicate how they are related to each other.

1. Analytic Solutions of Simple Cases of Constrained GPC

No analytical solution exists for the general case of constrained GPC. However, it is possible to calculate an analytical solution for some simple cases of constrained GPC. For most of the practical applications, a control horizon setting of $NU = 1$ or 2 can result in reasonably good control. This thesis presents an analytic solution of the rate and amplitude constrained SISO GPC for $NU = 1$ and 2 and MIMO GPC for $NU = 1$. Analytical solutions are much more computationally efficient than algorithmic solutions and furthermore they guarantee convergence.

Geometrically, for a control horizon of $NU = 2$, the quadratic cost function represents a set of concentric ellipses on the $\Delta u(t)$—$\Delta u(t+1)$ plane. Rate and amplitude constraints on the control signal can be respectively represented by a rectangle and a parallelogram on the $\Delta u(t)$—$\Delta u(t+1)$ plane. The feasible region for rate and amplitude constrained GPC is the intersection of the rectangle and the parallelogram, which can have a geometry of a 3 to 6 sided polygon. Geometrically, the solution of the constrained GPC is the point of contact of the smallest ellipse (lowest cost) with the feasible region on the $\Delta u(t)$—$\Delta u(t+1)$ plane.

2. Development of Software for Implementing Multivariable GPC

Software was developed for implementing the constrained multivariable generalized predictive control (MGPC) algorithm for processes with n outputs and m inputs (where n can be greater than or equal to or less than m). Both amplitude and rate constraints can be imposed on control signals and amplitude constraints can be imposed on process outputs.

The software can provide *dynamic* feedforward compensation for measurable disturbances. If a multivariable identification package is available, the software package can be implemented as an adaptive control algorithm. In this thesis, only SISO GPC was implemented as an adaptive algorithm.

The software solves the constrained cost function generated by GPC using a commercial quadratic programming software package, QPSOL [19]. QPSOL consists of a set of FORTRAN subroutines. The remaining part of the software is in 'C'. This software can be implemented in real-time.

3. Performance Evaluation of the Constrained GPC

The performance of constrained GPC algorithm was evaluated with the help of simulations and experimental runs. The algorithm was evaluated for setpoint decoupling, load rejection, with and without feedforward compensation, effects of model process mismatch, and effects of constraints. The SISO GPC was implemented as an adaptive control strategy while MGPC was implemented as a non-adaptive (fixed parameter) control strategy.

To evaluate the performance of non-adaptive multivariable GPC (MGPC), the control of the Shell heavy oil fractionator as documented in the Shell process control workshop [42] was simulated for different criteria. Many simulations were performed with non-adaptive MGPC to study the effects of model plant mismatch (MPM). Well tuned MGPC algorithm can handle a reasonable amount of MPM. It is shown that the output weighting (y_{wt}) plays an important role in tuning the MGPC algorithm for different amounts of MPM in different channels of the process. Adaptive MGPC has not been examined in this thesis. Simulations also demonstrate than the rate constraints on the control signal can stabilize systems which would otherwise be unstable.

Experimental runs conducted on a stirred tank heater, in SISO and MIMO configurations, and by Kwok [25] on a dog for controlling its mean arterial blood pressure, in SISO configuration, demonstrate that GPC is a powerful control algorithm capable of controlling fairly non-linear, open-loop unstable systems with time-varying delays and "optimally" decoupling interactions between various loops. The experimental runs also demonstrate that the performance of MGPC is far superior to multi-loop PID control.

The experimental runs demonstrate that constrained GPC has distinct advantages over the unconstrained GPC. The experimental runs show that rate constraints on the manipulated variables can stabilize poorly tuned systems. The experimental runs also show that tuning a highly non-linear process is non-trivial and the output weighting (y_{wt}) plays an important role in tuning the controller for multivariable processes with large model plant mismatch terms.

4. Scaling of Outputs and Manipulated Variables in Multivariable Systems

The simulations and experimental runs show that output scaling is an important tuning knob, in the presence of model process mismatch. This thesis suggests a general qualitative guideline for selecting the output weighting factors in a multivariable system, based on the results of simulation and experimental runs. The suggested guideline is:

In the presence of MPM, larger output weighting terms should be selected for the outputs of channels with higher model process mismatch.

The simulations and experimental runs support this statement.

This thesis describes the procedure for best scaling of a matrix in l_2-norm (Golub and Varah [20]). It is suggested that the manipulated variables also be scaled such that the condition number of the scaled system is minimized. This improves the numerical robustness of the process, without affecting the control law. Scaling of the manipulated variables in a multivariable systems with equal number of inputs and outputs has also been extended for systems with unequal number of inputs and outputs.

7.1 Recommendations for Future Work

1. The MGPC algorithm has been implemented as a non-adaptive control strategy. If a multivariable estimation package is available, the control algorithm can be implemented as a multivariable adaptive control algorithm.

2. The experimental runs and simulations demonstrate that output scaling is an important tuning knob. The guideline suggested for tuning this knob that large weighting should be given to outputs with larger model process mismatch is qualitative in nature. Mathematical treatment of the effects of output scaling can lead to a quantitative guideline and verify the recommendation for guideline for tuning the controller, in the presence of model plant mismatch.

References

[1] K. J. Åström and P. Ekhyoff, 1970. *System identification, a survey*, Report 7006, Lund Institute of Technology.

[2] K. J. Åström and B. Wittenmark, 1973. "On self-Tuning Regulators". *Automatica*, **9** (2) 185-199.

[3] K. J. Åström and B. Wittenmark, 1980. "Self-Tuning controllers–design principles and applications" *Applications of Adaptive Control*, ed. Narendra and Monopoli.

[4] K. J. Åström and B. Wittenmark, 1984. *Computer Controlled Systems: Theory And Design*, Prentice-Hall International Inc., 267-268.

[5] K. J. Åström and B. Wittenmark, 1989. *Adaptive Control*, Addison-Wesley, Reading, Mass.

[6] G. J. Bierman, 1977. *Factorisation Methods for Discrete-time Identification*, Academic Press, New York.

[7] D. Bonvin and D. A Mellichamp, 1985. "Rescaling State and Input Variables for the Purpose of Analyzing Process Models", *Proc. of 1985 ACC*, 1, 626-631.

[8] D. Bonvin and D. A Mellichamp, 1987. "A Scaling Procedure for the Structural and Interaction Analysis of Dynamic Models", *AICHE Journal*, **33**, (2), 250-257.

[9] D. W. Clarke and P. J. Gawthrop, 1975. "Self-tuning Controller" *Proc. IEE* **122** (9) 929-934.

[10] D. W. Clarke and P. J. Gawthrop, 1979. "Self-tuning Controller" *Proc. IEE* **126** (6) 633-640.

[11] D. W. Clarke, 1984. "Self-tuning Control of Nonminimum-Phase Systems" *Automatica* **20** (5) 501-517.

[12] D. W. Clarke, P. S. Tuffs and C. Mohtadi, 1985. "Self-tuning Control of a Difficult Process", *7th IFAC Symposium on Identification and System Parameter Estimation*, York.

[13] D. W. Clarke, C. Mohtadi and P. S. Tuffs, 1987. "Generalised Predictive Control — Part I. The Basic Algorithm" *Automatica* **23** (2) 137-148.

[14] D. W. Clarke, C. Mohtadi and P. S. Tuffs, 1987. "Generalised Predictive Control — Part II. Extensions and Interpretations" *Automatica* **23** (2) 149-160

[15] C. R. Cutler and B. L. Ramaker, 1980. "Dynamic Matrix Control – a Computer Control Algorithm" *Proc. of the 1980 Joint Automatic Control Conference*, San Francisco, U.S.A.

[16] R. M. C. De Keyser and A. R. Van Cauwenberghe, 1982. "Simple self-tuning multistep predictors", *Proc. of the 6th IFAC Symposium on Identification and System Parameter Estimation*, Washington D.C. U.S.A.

[17] R. M. C. De Keyser and A. R. Van Cauwenberghe, 1985. "Extended prediction self-adaptive control", *Proc. of the 7th IFAC Symposium on Identification and System Parameter Estimation*, York, U.K.

[18] P. E. Gill, W. Murray and M. H. Wright, 1981. *Practical Optimization*, Academic Press, New York.

[19] P. E. Gill, W. Murray, M. A. Saunders and M. H. Wright, 1984. "User's Guide for QPSOL: A Fortran Package for Quadratic Programming", *Technical Report SOL 84-6*, Dept. of Operations Research, Stanford University, Stanford, U.S.A.

[20] G. H. Golub and J. M. Varah, 1974. "On a characterization of the best l_2-scaling of a matrix", *SIAM J. Numer. Anal.*, **11**, 472-479.

[21] R. D. Johnston and G. W. Borton, 1984. "Improved Process Conditioning Using Internal Control Loops", *Int. J. Control*, **40**, No. 6, 1051-1063.

[22] R. D. Johnston and G. W. Borton, 1987. "Design and Performance Assessment of Control Systems Using Singular-Value Analysis", *Ind. Eng. Chem. Res.*, **26**, 830-839.

[23] R. E. Kalman, 1958. "Design of a self-optimizing control system", *Trans. ASME*, **80**, 468-478.

[24] J. P. Keller and D. Bonvin, 1987. "Selection of Inputs For The Purpose Of Model Reduction and Controller Design", *Proc. of the 10th IFAC Triennial World Congress*, Munich, FRG, 209-214.

[25] K. Y. Kwok, *Adaptive Blood Pressure Regulation*, Ph.D thesis in progress, Dept. of Chem. Eng., University of Alberta.

[26] R. Kulhavy, 1987. "Restricted Exponential Forgetting Real-time Identification", *Automatica* **23** (5) 589-600

[27] E. P. Lambert, 1987. *Process Control Applications of Long Range Prediction*, D.Phil. Thesis, University of Oxford.

[28] E. Lau, 1990. *Evaluation of Predictive Controllers for Processes with Time Delays*, M.Sc. Thesis, Dept. of Chem. Eng., University of Alberta.

[29] H. Lau and K. F. Jensen, 1985. "Evaluation of Changeover Control Policies by Singular Value Analysis—Effects of Scaling", *AICHE Journal*,**31**, (1), 135-146.

[30] A. R. McIntosh, 1988. *Performance and Tuning of Adaptive Generalized Predictive Control*, M.Sc. Thesis, Dept. of Chem. Eng., University of Alberta.

[31] A. R. McIntosh, S. L. Shah and D. G. Fisher, 1988. "Experimental Evaluation of Adaptive Control in the Presence of Disturbances and Model-Plant Mismatch", *Adaptive Control Strategies for Industrial Use*, S. L. Shah and G. Dumont, eds., Springer-Verlag Lecture Notes in Control and Information Sciences, Vol. 137, Springer-Verlag, New York. 145-174

[32] B. C. Moore, 1981. "Principal Component Analysis in Linear Systems: Controllability, Observability, and Model Reduction", *IEEE Transactions On Automatic Control*, **AC-26**, No. 1, 17-32.

[33] C. Moore, 1986. "Application of Singular Value Decomposition to the Design, Analysis, And Control of Industrial Processes", *Proc. of 1986 ACC*, Seatle, **2**, 643-650.

[34] P. R. Maurath, D. E. Seborg and D. A. Mellichamp, 1986. "Achieving Decoupling with Predictive Controllers", *Proc. of 1986 ACC*, Seatle, **3**, 1372-1377.

[35] C. Mohtadi, 1987. *Advanced Self-Tuning Algorithms* D.Phil thesis, University of Oxford.

[36] C. Mohtadi, 1988. "On the Role of Prefiltering in Parameter Estimation and Control", *Adaptive Control Strategies for Industrial Use*, S. L. Shah and G. Dumont, eds., Springer-Verlag Lecture Notes in Control and Information Sciences, Vol. 137, Springer-Verlag, New York. 121-144.

[37] A. J. Morris, Y. Nazer and R. K. Wood, 1981. "Multivariate Self-tuning Controllers for Distillation Column Control" *Proc. of 6th IFAC/IFID Conference on Digital Computer Applications to Process Control*, R. Iserman and H. Kaltenecker eds., Dusseldorf, F. R. Germany, 345-354.

[38] E. Mosca, G. Zappa and C. Manfredi, 1984. "Multistep Horizon Self-tuning controllers: the MUSMAR approach" *Proc. of the 9th IFAC World Congress*, Budapest, Hungary.

[39] A. Niederlinski, 1984. "A new look at Least-Squares Dynamic System Identification", *Int. J. Control*, **40**, No. 3, 467-478.

[40] V. Peterka, 1984. "Predictor based self-tuning control", *Automatica*, **20** (1), 39-50.

[41] M. J. D. Powell, 1989. "TOLMIN: A Fortran Package for Linearly Constrained Optimization Calculations", *Numerical Analysis Reports DAMTP 1989/NA2*, Dept. of Applied Mathematics and Theoretical Physics, University of Cambridge, Cambridge.

[42] D. M. Prett and M. Morari, 1986. *Shell Process Control Workshop*, Butterworth Publishers, Stoneham, MA.

[43] Z. Qiu, 1988. *MULTICON user manual*, University of Alberta.

[44] J. Richalet, A. Rault, J. L. Testud and J. Papon, 1978. "Model Predictive Heuristic Control: Applications to Industrial Processes" *Automatica*,**14** (5), 413-428.

[45] D. E. Seborg, T. F. Edgar and S. L. Shah, 1986. "Adaptive Control Strategies for Process Control: A Survey", *AICHE Journal*, **32** No.6, 881-913.

[46] S. L. Shah, C. Mohtadi and D. W. Clarke, 1987. "Multivaraible adaptive control without a prior knowledge of the delay matrix", *Systems & Control Letters*, **9**, 295-306, North-Holland.

[47] D. S. Shook, 1990. *Implementation Issues in Adaptive GPC*, Ph.D Thesis in Progress, Dept. of Chem. Eng., University of Alberta.

[48] D. S. Shook, C. Mohtadi and S. L. Shah, 1990. "Process Identifiation for Long Range Predictive Control", *IEE Proc. D*, In press, 1990.

[49] D. S. Shook, C. Mohtadi and S. L. Shah, 1990. "Adaptive Filtering And GPC", *Proc. of 1990 ACC*, San Diego.

[50] M. G. Singh and A. Titli, 1978. *Systems Decomposition, Optimisation and Control*, Pergamon Press.

[51] S. Skogestad, M. Morari and J. C. Doyle, 1988. "Robust Control of Ill-Conditioned Plants: High-Purity Distillation," *IEEE Transactions on Automatic Control*, , **33** (12).

[52] O. J. M. Smith, 1957. "Closer Control of Loops with Dead Time", *Chem. Eng. Prog.*, **53** (5), 217-219.

[53] Sripada N. Rao, 1988. *Multi-Step Adaptive Predictive Control with Disturbance Modeling*, Ph.D. Thesis, Department of Chemical Engineering, University of Alberta, 161-164.

[54] G. Stephanopoulos, 1984. *Chemical Process Control*, Prentice-Hall International Inc., New Jersey.

[55] T. T. C. Tsang and D. W. Clarke, 1988. "Generalised Predictive Control with Input Constraints", *IEE Proc. D*, **135**, (6), 451-460

[56] P. S. Tuffs, 1984. *Self-Tuning Control: Algorithms and Applications* D.Phil thesis, University of Oxford.

[57] B. Wahlberg and L. Ljung, 1986. "Design Variables for Bias Distribution in Transfer Function Estimation", *IEEE Trans. Auto. Cont*, **AC-31** (2) 134-144

[58] P. E. Wellstead, J. M. P. Edmunds, D. Prager, and P. Zanker, 1979. "Self-tuning pole/zero assignment regulators" *Int. J. Control*, **30** (1) 1-26.

[59] B. E. Ydstie, 1982. *Robust Adaptive Control of Chemical Processes*, Ph.D Thesis, Department of Chemical Engineering and Chemical Technology, Imperial College, London, U.K.

[60] B. E. Ydstie, 1984. "Extended horizon Adaptive Control", *Proc. of 9th IFAC World Congress*, Budapest, Hungary.

[61] *QNX Reference Guide, Version 2.1*, Quantum Software Systems Ltd., Kanata, Ontario K2M 1W8, Canada.

[62] Opto 22 Inc. Ltd, 15461 Springdale Street, Huntington Beach, CA 92649.

Appendix A

Analytical Solution of Some Cases of Constrained GPC

A.1 Analytical Solution of SISO GPC for $NU = 2$

The possible optimal solutions for the analytical solution of rate and amplitude constrained SISO GPC for $NU = 2$ are shown in this section. The notation of section 3.1.3 is used in describing the solution.

Case 0: No constraints violated

$$\begin{aligned} \Delta u(t) &= \frac{h_{22}c_1 - h_{12}c_2}{h_{12}^2 - h_{11}h_{22}} \\ \Delta u(t+1) &= \frac{h_{11}c_2 - h_{12}c_1}{h_{12}^2 - h_{11}h_{22}} \end{aligned}$$

Sufficient conditions for optimality : $a_i \geq 0 \quad \forall \quad i = 1$ to 6

Case 1: If optimal solution is on a_1, then

$$\begin{aligned} \Delta u(t) &= \alpha_1 \\ \Delta u(t+1) &= -\frac{c_2 + h_{12}\alpha_1}{h_{22}} \end{aligned}$$

Sufficient conditions for optimality: $a_i \geq 0 \quad \forall \quad i = 2$ to 6; $\mu_1 < 0$

Case 2: If optimal solution is on a_2, then

$$\Delta u(t) = \beta_1$$

$$\Delta u(t+1) = -\frac{c_2 + h_{12}\beta_1}{h_{22}}$$

Sufficient conditions for optimality: $a_i \geq 0 \quad \forall \quad i = 1,3,4,5,6; \quad \mu_2 < 0$

Case 3: If optimal solution is on a_3, then

$$\Delta u(t) = -\frac{\alpha_2(h_{12} - h_{22}) + c_1 - c_2}{h_{11} - 2h_{12} - h_{22}}$$
$$\Delta u(t+1) = \frac{\alpha_2(h_{11} - h_{12}) + c_1 - c_2}{h_{11} - 2h_{12} - h_{22}}$$

Sufficient conditions for optimality: $a_i \geq 0 \quad \forall \quad i = 1,2,4,5,6; \quad \mu_3 < 0$

Case 4: If optimal solution is on a_4, then

$$\Delta u(t) = -\frac{\beta_2(h_{12} - h_{22}) + c_1 - c_2}{h_{11} - 2h_{12} - h_{22}}$$
$$\Delta u(t+1) = \frac{\beta_2(h_{11} - h_{12}) + c_1 - c_2}{h_{11} - 2h_{12} - h_{22}}$$

Sufficient conditions for optimality: $a_i \geq 0 \quad \forall \quad i = 1,2,3,5,6; \quad \mu_4 < 0$

Case 5: If optimal solution is on a_5, then

$$\Delta u(t) = -\frac{h_{12}\alpha_3 + c_1}{h_{11}}$$
$$\Delta u(t+1) = \alpha_3$$

Sufficient conditions for optimality: $a_i \geq 0 \quad \forall \quad i = 1,2,3,4,6; \quad \mu_5 < 0$

Case 6: If optimal solution is on a_6, then

$$\Delta u(t) = -\frac{h_{12}\beta_3 + c_1}{h_{11}}$$
$$\Delta u(t+1) = \beta_3$$

Sufficient conditions for optimality: $a_i \geq 0 \quad \forall \quad i = 1,2,3,4,5; \quad \mu_6 < 0$

Case 7: If optimal solution is at intersection of a_1 and a_3, then

$$\Delta u(t) = \alpha_1$$
$$\Delta u(t+1) = \alpha_2 - \alpha_1$$

Sufficient conditions for optimality: $a_i \geq 0 \ \forall \ \ i = 2,4,5,6; \ \ \mu_1, \mu_3 < 0$

Case 8: If optimal solution is at intersection of a_1 and a_4, then

$$\begin{aligned} \Delta u(t) &= \alpha_1 \\ \Delta u(t+1) &= \beta_2 - \alpha_1 \end{aligned}$$

Sufficient conditions for optimality: a [illegible] $\forall \ \ i = 2,3,5,6; \ \ \mu_1, \mu_4 < 0$

Case 9: If optimal solution is at intersection of a_1 and a_5, then

$$\begin{aligned} \Delta u(t) &= \alpha_1 \\ \Delta u(t+1) &= \alpha_3 \end{aligned}$$

Sufficient conditions for optimality: $a_i \geq 0 \ \forall \ \ i = 2,3,4,6; \ \ \mu_1, \mu_5 < 0$

Case 10: If optimal solution is at intersection of a_1 and a_6, then

$$\begin{aligned} \Delta u(t) &= \alpha_1 \\ \Delta u(t+1) &= \beta_3 \end{aligned}$$

Sufficient conditions for optimality: $a_i \geq 0 \ \forall \ \ i = 2,3,4,5; \ \ \mu_1, \mu_6 < 0$

Case 11: If optimal solution is at intersection of a_2 and a_3, then

$$\begin{aligned} \Delta u(t) &= \beta_1 \\ \Delta u(t+1) &= \alpha_2 - \beta_1 \end{aligned}$$

Sufficient conditions for optimality: $a_i \geq 0 \ \forall \ \ i = 1,4,5,6; \ \ \mu_2, \mu_3 < 0$

Case 12: If optimal solution is at intersection of a_2 and a_4, then

$$\begin{aligned} \Delta u(t) &= \beta_1 \\ \Delta u(t+1) &= \beta_2 - \beta_1 \end{aligned}$$

Sufficient conditions for optimality: $a_i \geq 0 \ \forall \ \ i = 1,3,5,6; \ \ \mu_2, \mu_4 < 0$

Case 13: If optimal solution is at intersection of a_2 and a_5, then

$$\begin{aligned}\Delta u(t) &= \beta_1\\ \Delta u(t+1) &= \alpha_3\end{aligned}$$

Sufficient conditions for optimality: $a_i \geq 0 \;\; \forall \;\; i = 1,3,4,6; \;\; \mu_2, \mu_5 < 0$

Case 14: If optimal solution is at intersection of a_2 and a_6, then

$$\begin{aligned}\Delta u(t) &= \beta_1\\ \Delta u(t+1) &= \beta_3\end{aligned}$$

Sufficient conditions for optimality: $a_i \geq 0 \;\; \forall \;\; i = 1,3,4,5; \;\; \mu_2, \mu_6 < 0$

Case 15: If optimal solution is at intersection of a_3 and a_5, then

$$\begin{aligned}\Delta u(t) &= \alpha_2 - \alpha_3\\ \Delta u(t+1) &= \alpha_3\end{aligned}$$

Sufficient conditions for optimality: $a_i \geq 0 \;\; \forall \;\; i = 1,2,4,6; \;\; \mu_3, \mu_5 < 0$

Case 16: If optimal solution is at intersection of a_3 and a_6, then

$$\begin{aligned}\Delta u(t) &= \alpha_2 - \beta_3\\ \Delta u(t+1) &= \beta_3\end{aligned}$$

Sufficient conditions for optimality: $a_i \geq 0 \;\; \forall \;\; i = 1,2,4,5; \;\; \mu_3, \mu_6 < 0$

Case 17: If optimal solution is at intersection of a_4 and a_5, then

$$\begin{aligned}\Delta u(t) &= \beta_2 - \alpha_3\\ \Delta u(t+1) &= \alpha_3\end{aligned}$$

Sufficient conditions for optimality: $a_i \geq 0 \;\; \forall \;\; i = 1,2,3,6; \;\; \mu_4, \mu_5 < 0$

Case 18: If optimal solution is at intersection of a_4 and a_6, then

$$\begin{aligned}\Delta u(t) &= \beta_2 - \beta_3 \\ \Delta u(t+1) &= \beta_3.\end{aligned}$$

Sufficient conditions for optimality: $a_i \geq 0 \;\; \forall \;\; i = 1, 2, 3, 5; \;\; \mu_4, \mu_6 < 0$

Case 19: No feasible solution due to inconsistent constraints.

A.2 Analytical Solution of MIMO (2×2) GPC for $NU = 1$

This section gives the solution of rate and amplitude constrained GPC for a 2×2 MIMO system. The solution is based on the algorithm developed in section 3.1.5 for a n-output, m-input system and uses the same notation.

For $NU = 1$, the equation 3.28 can be rewritten as:

$$\begin{aligned} 2(h_{11}\Delta u_1(t) + h_{12}\Delta u_2(t) + c_1) + \mu_{l1} - \mu_{u1} = 0 \\ 2(h_{12}\Delta u_1(t) + h_{22}\Delta u_2(t) + c_2) + \mu_{l2} - \mu_{u2} = 0 \end{aligned} \tag{A.1}$$

where

$$\begin{aligned} \mathcal{G}^T\mathcal{G} + \Lambda &= H_{m \times m} &= \begin{bmatrix} h_{11} & h_{12} \\ h_{12} & h_{22} \end{bmatrix} \\ (\mathbf{f} - \mathbf{w})^T\mathcal{G} &= c^T &= [c_1 \quad c_2] \end{aligned} \tag{A.2}$$

The possible solutions for the rate and amplitude constrained MIMO (2×2) GPC for $NU = 1$ are shown below:

Case 0: No constraints violated

$$\begin{aligned}\Delta u_1(t) &= \frac{h_{22}c_1 - h_{12}c_2}{{h_{12}}^2 - h_{11}h_{22}} \\ \Delta u_2(t) &= \frac{h_{11}c_2 - h_{12}c_1}{{h_{12}}^2 - h_{11}h_{22}}\end{aligned}$$

Sufficient conditions for optimality: $a_{l1}, a_{u1}, a_{u1}, a_{u2} \geq 0$

Case 1: If optimal solution is on a_{l1}, then

$$\begin{aligned} \Delta u_1(t) &= \alpha_1 \\ \Delta u_2(t) &= -\frac{c_2 + h_{12}\alpha_1}{h_{22}} \end{aligned}$$

Sufficient conditions for optimality: $a_{l2}, a_{u1}, a_{u2} \geq 0 \quad \mu_{l1} < 0$

Case 2: If optimal solution is on a_{u1}, then

$$\begin{aligned} \Delta u_1(t) &= \beta_1 \\ \Delta u_2(t) &= -\frac{c_2 + h_{12}\beta_1}{h_{22}} \end{aligned}$$

Sufficient conditions for optimality: $a_{l1}, a_{l2}, a_{u2} \geq 0 \quad \mu_{u1} < 0$

Case 3: If optimal solution is on a_{l2}, then

$$\begin{aligned} \Delta u_1(t) &= -\frac{h_{12}\alpha_2 + c_1}{h_{11}} \\ \Delta u_2(t) &= \alpha_2 \end{aligned}$$

Sufficient conditions for optimality: $a_{l1}, a_{u1}, a_{u2} \geq 0 \quad \mu_{l2} < 0$

Case 4: If optimal solution is on a_{u2}, then

$$\begin{aligned} \Delta u_1(t) &= -\frac{h_{12}\beta_2 + c_1}{h_{11}} \\ \Delta u_2(t) &= \beta_2 \end{aligned}$$

Sufficient conditions for optimality: $a_{l1}, a_{l2}, a_{u1} \geq 0 \quad \mu_{u2} < 0$

Case 5: If optimal solution is at intersection of a_{l1} and a_{l2}, then

$$\begin{aligned} \Delta u_1(t) &= \alpha_1 \\ \Delta u_2(t) &= \alpha_2 \end{aligned}$$

Sufficient conditions for optimality: $a_{u1}, a_{u2} \geq 0 \quad \mu_{l1}, \mu_{l2} < 0$

Case 6: If optimal solution is at intersection of a_{l1} and a_{u2},then

$$\begin{aligned}\Delta u_1(t) &= \alpha_1 \\ \Delta u_2(t) &= \beta_2\end{aligned}$$

Sufficient conditions for optimality: $a_{l2}, a_{u1} \geq 0 \quad \mu_{l1}, \mu_{u2} < 0$

Case 7: If optimal solution is at intersection of a_{u1} and a_{l2}, then

$$\begin{aligned}\Delta u_1(t) &= \beta_1 \\ \Delta u_2(t) &= \alpha_2\end{aligned}$$

Sufficient conditions for optimality: $a_{l1}, a_{u2} \geq 0 \quad \mu_{u1}, \mu_{l2} < 0$

Case 8: If optimal solution is at intersection of a_{u1} and a_{u2}, then

$$\begin{aligned}\Delta u_1(t) &= \beta_1 \\ \Delta u_2(t) &= \beta_2\end{aligned}$$

Sufficient conditions for optimality: $a_{l1}, a_{l2} \geq 0 \quad \mu_{u1}, \mu_{u2} < 0$

Case 9: No feasible solution due to inconsistent constraints.

Appendix B

Proof of equivalence of unscaled and D_2 scaled MGPC algorithm

The purpose of this appendix is to show that the MGPC control law for the scaled and the unscaled system is invariant of D_2 for an infinite precision computing machine.

Equation 2.16 can be rewritten in the vector form as

$$\hat{Y} = \mathcal{G}\tilde{U} + \mathcal{F}$$

where

$$\begin{aligned}
\hat{Y} &= [\hat{\mathbf{y}}(t+N_1), \ldots, \hat{\mathbf{y}}(t+N_2)]^T \\
\tilde{U} &= [\Delta\mathbf{u}(t), \ldots, \Delta\mathbf{u}(t+NU-1)]^T \\
\mathcal{F} &= [\mathbf{f}(t+N_1) \;\ldots\; \mathbf{f}(t+N_2)]^T \\
\mathcal{G} &= \begin{bmatrix} G_{N_1-1} & \cdots & G_0 & 0 & 0 & \cdots & 0 \\ G_{N_1} & \cdots & G_1 & G_0 & 0 & \cdots & 0 \\ \vdots & \ddots & & & & & \vdots \\ G_{N_2-1} & \cdots & & & - & & G_{N_2-NU} \end{bmatrix}
\end{aligned}$$

where each G_i is a submatrix of dimension $n \times m$, the elements of which are the step responses of i_{th} sampling interval of each individual input-output The free response $\mathbf{f}(t+j)$ is the response of the plant assuming that future controls equal $\mathbf{u}(t-1)$ or that $\Delta\mathbf{u}(t+j) = 0 \quad (\forall j = 0, 1, 2, \ldots, NU-1)$. The dimensions of vectors $\hat{\mathbf{y}}(t+i), \mathbf{f}(t+i)$ and $\Delta\mathbf{u}(t+j)$ $\forall i = N_1$ to N_2, $j = 0$ to $(NU-1)$) are n, n and m respectively.

If vector $\mathbf{W} = [\mathbf{w}(t + N_1), \ldots, \mathbf{w}(t + N_2)]^T$ defines the setpoint trajectory, then the objective function of MGPC (equation 6.3) can be rewritten as:

$$J = [\mathcal{G}\tilde{U} + \mathcal{F} - \mathbf{W}]^T [\mathcal{G}\tilde{U} + \mathcal{F} - \mathbf{W}] \tag{B.1}$$

and the solution minimizing J to give optimal suggested control increment sequence is:

$$\tilde{U} = (\mathcal{G}^T \mathcal{G})^{-1} \mathcal{G}^T (\mathbf{W} - \mathcal{F}) \tag{B.2}$$

Now, consider the input scaled system with matrix D_2 (D_1 = identity matrix). All the submatrices G_i of $\mathcal{G}$ are post multiplied by the matrix D_2, i.e. the controller sees $G(s)D_2$ as the process. Define a block diagonal matrix D as:

$$D = \text{ diag } (D_2, D_2, \ldots, D_2)$$

i.e D = matrix of NU block diagonal D_2 matrices.

The relation between the unscaled and inputs scaled system are:

$$\mathcal{G}_s \;=\; \mathcal{G}D \tag{B.3}$$

$$\tilde{U} \;=\; D\tilde{U}_s \tag{B.4}$$

The optimal control sequence for the scaled system given by the control algorithm is:

$$\tilde{U}_s = (\mathcal{G}_s^T \mathcal{G}_s)^{-1} \mathcal{G}_s^T (\mathbf{W} - \mathcal{F}_s) \tag{B.5}$$

Let us assume that scaled control law (equation B.5) gives control action related to unscaled control law (equation B.2) related by equation (B.4). If this assumption is true, the free response for the scaled ($\mathcal{F}_S$) and unscaled system ($\mathcal{F}$) will be identical. To prove the assumption that the scaled control law (equation B.5) gives control action related to the unscaled control law (equation B.2) by equation (B.4), consider equation (B.5):

$$\tilde{U}_s \quad = (\mathcal{G}_s^T \mathcal{G}_s)^{-1} \mathcal{G}_s^T (\mathbf{W} - \mathcal{F})$$

$$\begin{aligned}
\Rightarrow\quad \hat{U}_s &= ((\mathcal{G}D)^T\mathcal{G}D)^{-1}(\mathcal{G}D)^T(\mathbf{W}-\mathcal{F}) \\
\Rightarrow\quad \hat{U}_s &= D^{-1}(\mathcal{G}^T\mathcal{G})^{-1}\mathcal{G}^T(\mathbf{W}-\mathcal{F}) \\
\Rightarrow\quad D\hat{U}_s &= \hat{U}
\end{aligned}$$

Which confirms equation (B.4). This proves that the control law with unscaled inputs and scaled inputs is identical. The control trajectory to the physical system should be same for both the cases, if the numerical computation is of infinite precision.

The above proof assumes that the control weighting matrix $\Lambda(j)$ in equation (6.3) is a zero. For a non-zero $\Lambda(j)$ in equation (6.3) of the unscaled system, the control law will be unchanged. If $\Lambda_s(j) = D^T\Lambda(j)D$ is used as the control weighting matrix of the scaled system then the control law for the scaled system is given by:

$$\hat{U}_s = (\mathcal{G}_s^T\mathcal{G}_s + \Lambda_s)^{-1}\mathcal{G}_s^T(\mathbf{W}-\mathcal{F}_s) \tag{B.6}$$

Appendix C

Listing of the Software for Implementing MGPC

This appendix lists the code implemented under MATLAB to simulate multivaraible Generalized Predictive Control Algorithm. The algorithm implemented under QNX operating system also has a similar structure.

Script file: init_mv.m

```
%     script file:::  INIT_MV
%
%     THIS SCRIPT FILE DEFINES THE TRANSFER FUNCTIONS, MGPC CONTROLLER
%     SETTINGS FOR CALCULATING THE CONTROL ACTION.  THIS FILE IS
%     CURRENTLY SET FOR A 3-INPUT 3-OUTPUT SYSTEM.  ONLY THIS FILE
%     IS DEPENDENT ON THE ORDER OF THE MV SYSTEM AND A PART OF THE
%     PROGRAM SHOULD BE MODIFIED FOR SYSTEMS OF DIFFERENT NUMBER
%     OF INPUTS AND OUTPUTS.  THE REST OF THE MATLAB mimo_gpc
%     PROGRAM IS INDEPENDENT OF NUMBER OF INPUTS AND OUTPUTS.
%
%        FROM THE TRANSFER FUNCTION DESCRIPTION, THIS FILE SHOULD
%     CALCULATE THE CARIMA MODEL FOR THE MULTIVARIABLE SYSTEM.
%     THIS FUNCTION SHOULD GENERATE THE MIMO STEP RESPONSE MATRIX, G.
%
%        SOME OF THE MAIN VARAIBLES FOR DEFINING THE PROCESS ARE
%
%          n      number of outputs  (CURRENTLY SET AS 3)
%          m      number of inputs   (CURRENTLY SET AS 3)
%          N1     minimum costing horizon
%          N2     maximum costing horizon
%          ts     sampling rate
%          den    denominator polynomial (s-domain)
%          num    numerator polynomial (s-domain).
%          td     delay matrix
%          Q      matrix the weightage for each of control
%                               outputs (lambda of SISO)
%          T      filter T polynomial
%          conmat The constraint matrix
%
%
%          p      no. of element in each row of A.
%          q      no. of elements in each Bij numerator polynomial
%  note::::  p,q are important variables used for maintaining the size
%            of the matrices and controlling the indices; for eg.
%           Bij = dn11 zeroes(q-length(dn11)) i.e. all Bij are of length q.
%
%    THE OTHER IMPORTANT VARIABLES WHICH WILL BE USED IN OTHER FILES OF
%    THIS ALGORITHM ARE:
%          du    manipulated variable 'DELTA U'
%          y     process output
%          duf   T-filtered manipulated variable 'DELTA U'
%          yf    T-filtered process output
%          uu    the absolute value of the implemented signal
%          f     free-response of the process
%          E,F   E matrix and F vector as required by matlab function
%                   for defineing the cost to be minimized.
%
%   Rajendra Mutha;  July 4, 90.

clear;   % CLEAR ALL THE PREVIOUSLY DEFINED VARIABLES

n=3;                                % NO. OF OUTPUTS OF THE MIMO SYSTEM
m=3;                                % NO. OF INPUTS OF THE MIMO SYSTEM
setpt=[10,10,10;  20,20,20]';       % SETPOINTS AS STEP CHANGES
step_size = 25;                     % CHANGE THE SETPOINT EVERY step_size
ITER = 30;                          % TOTAL NO. OF SIMULATION ITERATIONS.
T=[1     -.8  ];                    % T-filter
N1=1;  N2=15;  NU=2;                % GPC TUNING KNOBS
Q(1,:)=[.1 .1 .1];                  % CONTROL WEIGHTING (LAMBDA WEIGHTING);
ts=0.1;                             % DISCRETIZING TIME FOR TRANS.FN. FROM s TC z
                                    %                  s TO z DOMAIN

%     NOW DEFINE THE TRANSFER FUNCTION OF MIMO PROCESS IN s-DOMAIN AS
%     NUMERATOR AND DENOMINATOR VECTORS AND SCALAR DELAY:

num11=6    ;                         den11= [4  1];            td(1,1)=  0;
num12=3    ;                         den12= [5  1];            td(1,2)= 0;
num13=2    ;                         den13= [20 1];            td(1,3)= 2;

num21=.7   ;                         den21= [1 1];             td(2,1)= 2;
num22=1    ;                         den22= [1 1];             td(2,2)= 1;
num23=5    ;                         den23= [32 1];            td(2,3)=0;

num31=1    ;                         den31= [24 1];            td(3,1)= 2;
num32=4.5  ;                         den32= [1 1];             td(3,2)= 0;
num33=2    ;                         den33= [11 1];            td(3,3)= 1;


%    DEFINE THE LOWER CONSTRAINTS AND UPPER CONSTRAINTS ON 'U' AND 'DELTA U'
L= [-1000;  -10000 ; -10000 ];U= [10000 ;10000 ; 10000];    % U CONSTRAINTS
DL= [-1000;  -10000 ; -10000 ];DU= [10000 ;10000 ; 10000]; % du CONSTRAINTS

%   THIS COMPLETES  DEFINING THE PROBLEM.  NOW THIS FILE COMPUTES
%   THE CARIMA MODEL, STEP RESPONSE MATRIX, THE LOWER AND UPPER BOUND
%   OF THE CONSTRAINTS, THE CONSTRAINT MATRIX.
%**********************************************************************
for i=1:NU,tempQ=[Q(1,:).^0.5,tempQ];end;  %TO POWER 0.5 AS ||EX-F|| IN FUNCTION
Q=diag( tempQ);                            % lsi  REQUIRES Q^0.5

% DISCRETIZE THE TRANSFER FUNCTIONS.

[dn11,dd11,td(1,1)]=stoz(num11,den11,ts,td(1,1));
```

```
[dn12,dd12,td(1,2)]=stoz(num12,den12,ts,td(1,2));
[dn13,dd13,td(1,3)]=stoz(num13,den13,ts,td(1,3));
[dn21,dd21,td(2,1)]=stoz(num21,den21,ts,td(2,1));
[dn22,dd22,td(2,2)]=stoz(num22,den22,ts,td(2,2));
[dn23,dd23,td(2,3)]=stoz(num23,den23,ts,td(2,3));
[dn31,dd31,td(3,1)]=stoz(num31,den31,ts,td(3,1));
[dn32,dd32,td(3,2)]=stoz(num32,den32,ts,td(3,2));
[dn33,dd33,td(3,3)]=stoz(num33,den33,ts,td(3,3));

%     CALCULATE THE STEP RESPONSES OF THE PROCESS FOR FORMING THE
%     STEP RESPONSE MATRIX.

step11=dstep(dn11,dd11,1+N2);      step11=step11(2:N2+1);
step12=dstep(dn12,dd12,1+N2);      step12=step12(2:N2+1);
step13=dstep(dn13,dd13,1+N2);      step13=step13(2:N2+1)
step21=dstep(dn21,dd21,1+N2);      step21=step21(2:N2+1);
step22=dstep(dn22,dd22,1+N2);      step22=step22(2:N2+1);
step23=dstep(dn23,dd23,1+N2);      step23=step23(2:N2+1);
step31=dstep(dn31,dd31,1+N2);      step31=step31(2:N2+1);
step32=dstep(dn32,dd32,1+N2);      step32=step32(2:N2+1);
step33=dstep(dn33,dd33,1+N2);      step33=step33(2:N2+1);

%     FORM THE STEP RESPONSE MATRIX
G=[step11,step12,step13
   step21,step22,step23
   step31,step32,step33];

%     FORM THE A_TILDE = A * DELTA IN CARIMA MODEL
%     THE STRUCTURE OF A_TILDE SHOULD BE A DIAGONAL MATRIX WITH EACH
%     OF THE DIAGONAL ELEMENT BEING A VECTOR.
%     IN THIS PROGRAM, ALL THE ZEROS ARE NOT SHOWN FOR BREVITY AND
%     EFFICIENCY.  THE MATRIX IS SHOWN AS A MATRIX OF TWO DIMENSION
%     WITH EACH ROW REPRESENTING THE DIAGONAL ELEMENT VECTOR.

tem=[1 -1];                                 % DELTA
a1=conv(tem,conv(dd11,conv(dd12,dd13)));
a2=conv(tem,conv(dd21,conv(dd22,dd23)));
a3=conv(tem,conv(dd31,conv(dd32,dd33)));

%     DEFINE p AS THE MAXIMUM SIZE OF THE VECTOR OF ELEMENTS OF A_MATRIX
p=max(max(length(a1),length(a2)),length(a3))

%     SEPERATE THE PURE DELAYS AND ZOH FROM THE DISCRETIZED
%     NUMERATORS OF THE PROCESS.
dn11=dn11(td(1,1)+2:length(dn11));   dn12=dn12(td(1,2)+2:length(dn12));
dn13=dn13(td(1,3)+2:length(dn13));   dn21=dn21(td(2,1)+2:length(dn21));
dn22=dn22(td(2,2)+2:length(dn22));   dn23=dn23(td(2,3)+2:length(dn23));
dn31=dn31(td(3,1)+2:length(dn31));   dn32=dn32(td(3,2)+2:length(dn32));
dn33=dn33(td(3,3)+2:length(dn33));

%     FORMING THE B-MATRIX OF THE CARIMA MODEL BY CONVOLUTING THE
%     NUMERATORS WITH THE DENOMINATORS OF THE OTHERS.
%        THE CARIMA MODEL is:: Atilde*y = B*du
%  _                        _   _                                                  _
% | DELTA*dd11*dd12*dd13 |    | dn11*dd12*dd13  dd11*dn12*dd13  dd11*dd12*dn13 |
% | DELTA*dd21*dd22*dd23 |y =| dn21*dd22*dd23  dd21*dn22*dd23  dd21*dd22*dn23 |du
% |_DELTA*dd31*dd32*dd33_|    |_dn31*dd32*dd33  dd31*dn32*dd33  dd31*dd32*dn33_|
%
```

```
dn11=conv(dn11,conv(dd12,dd13));
dn12=conv(dn12,conv(dd11,dd13));
dn13=conv(dn13,conv(dd11,dd12));
dn21=conv(dn21,conv(dd22,dd23));
dn22=conv(dn22,conv(dd21,dd23));
dn23=conv(dn23,conv(dd21,dd22));
dn31=conv(dn31,conv(dd32,dd33));
dn32=conv(dn32,conv(dd31,dd33));
dn33=conv(dn33,conv(dd31,dd32));

q1=max(max(length(dn11),length(dn12)),length(dn13));
q2=max(max(length(dn21),length(dn22)),length(dn23));
q3=max(max(length(dn31),length(dn32)),length(dn33));
q=max(max(q1,q2),q3);
q=max(q,3);     % A MINIMUM SIZE OF 3 IS FOR IMPLEMENTING T-FILTER (?)

 y=zeros(n,p);
 maxdel=max(max(td));
 du=zeros(m,maxdel+q);

Atilde=[a1 zeros(1,p-length(a1));a2 zeros(1,p-length(a2));a3 zeros(1,p-length(a3))];

B=[
dn11 zeros(1,q-length(dn11)) dn12 zeros(1,q-length(dn12)) dn13 zeros(1,q-length(dn13))
dn21 zeros(1,q-length(dn21)) dn22 zeros(1,q-length(dn22)) dn23 zeros(1,q-length(dn23))
dn31 zeros(1,q-length(dn31)) dn32 zeros(1,q-length(dn32)) dn33 zeros(1,q-length(dn33)) ];



% GENERATE THE CONSTRAINT MATRIX AS REQUIRED BY THE m-FILE 'LSI.M'
 for i=1:NU,
      for j=1:NU,
         if(j<=i), conmat1((i-1)*m+1:i*m,(j-1)*m+1:j*m)=eye(m);end;
         if (j==i),conmat2((i-1)*m+1:i*m,(j-1)*m+1:j*m)=eye(m);end;
      end
end
conmat=[conmat1;-conmat1;conmat2;-conmat2];  %CONSTRAINT ORDER  L,-U,DL,-DU
```

Function: stoz.m

```
function [bz,az,zd]=stoz(bs,as,h,td)
%
%[bz,az,zd]=stoz(bs,as,h,td)
%     Convert s domain SISO transfer function to z domain.
%     It can deal with fractional dead-times.
%
%     bs    Zero polynomial
%     as    Pole polynomial
%     h     sample-time
%     td    time delay
%     bz    z-domain Zero polynomial (includes leading zeros for delay)
%     az    z-domain Pole polynomial
%     zd    Units of discrete time delays (represented seperately also);
%
x=td/h; k=fix(x); delta=td-k*h;
[phis,gams,deltas,omegas]=tf2ss(bs,as);
[phi,gam]=c2d(phis,gams,h);
```

```
 if delta > 0;
   [phi2,gam2]=c2d(phis,gams,delta);
   hd=h-delta;
   [phi3,gam3]=c2d(phis,gams,hd);
   b2=phi3*gam2;
   [bz1,az]=ss2tf(phi,gam3,deltas,omegas,1);
   [bz2,az]=ss2tf(phi,b2,deltas,omegas,1);
   bz=[bz1 0]+[0 bz2];
 else;
   [bz,az]=ss2tf(phi,gam,deltas,omegas,1);
 end;
 if k>0, bz=[zeros(1,k) bz];end;
   zd=k;
 return
```

Function: mcaly.m

```
function [output]=mcaly(Atilde,B,y,du,m,n,p,q,maxdel,td)
%
%     [output]=MCALY(Atilde,B,y,du,m,n,p,q,maxdel,td)
%     THIS FUNCTION GENERATES THE OUTPUT y(t) FOR A MIMO SYSTEM GIVEN
%     A CARIMA MODEL (STRUCTURE DEFINED IN FUNCTION 'init_mv.m'),
%     GIVEN [y(t-1) y(t-2) ...] AND [du(t-1) du(t-2), ...]
%
%     Atilde matrix having diagonoal elements of Atilde*Y = B del(U), in (q-1)
%     B      matrix of Atilde*Y=B del(U) in terms of (q-1)
%     y      previous plants output from y(t-1)
%               to y(t-p);  matrix order  (n x p)
%     du     previous controller outputs from u(t-1)
%                to du(t-maxdelay-q);   matrix order (m x (p+maxdel))
%     maxdel maximum delay of the system (scalar)
%     td     delay matrix (n x m)
%     output plant output at sampling interval t.  This is calculated given
%                du(t-i) and  y(t-j) ,  where i,j = 1,2,...
%
%     p      order of maximum numerator polynomial in A + 1;
%     q      same as p, but for matrix B;        /* refer to fn. ININIT_MV.M */

%     Rajendra Mutha;  July 11, 89.


for i=1:n,
temp1=0;temp2=0;
      for pp=1:p-1,
         temp1=temp1+ Atilde(i,pp+1)*y(i,pp);
      end;
      for j=1:m,
         for qq=1:q
            temp2=temp2+B(i,(j-1)*q+qq)*du(j,td(i,j)+qq);
         end;
      end;
      output(i)=(temp2-temp1)/Atilde(i,1);
end;
output=output(:);
return;
```

Function: mgeng.m

```
function [g] =mgeng(G,N1,N2,NU,Q,n,m)
%
%     [g]  = MGENG(G,N1,N2,NU,Q,n,m)
%     This function will generate the g matrix from the step response
%     matrix G, in the form required by function 'lsi' of MATLAB.
%
%     n      n ouput system
%     m      m input system
%     G      The step responses of the transfer functions of the
%               MIMO system (n-output  m-input).  The step response
%               of each transfer function is from 1 to N2 instance
%               of time.
%     Q      Control weight factor diagonal matrix (NU*m x NU*m);
%               same as lamda in SISO GPC.
%     N1,N2 Costing horizon
%     NU    Controlihorizon;
%
%  Rajendra Mutha;  July 4, 90.

%     REFORMAT THE STEP RESPONSE MATRIX, G, INTO MATRIX gtemp SUCH THAT
%     THE iTH SUBMATRIX REPRESENTS THE (N2-i)TH STEP RESPONSE OF
%     ALL THE n BY m PROCESSES.  THE MATRIX gtemp CONSIST OF N2 STEP
%     RESPONSES AND IS APPENDED AT THE END WITH SUBMATRICES OF ZEROS.
%
%       gtemp  =  [g(N2) g(N2-1) ..g(j)..  g(2) g(1) 0 0 .. 0];
%
%       (  The index of first column of g(j) is (N2-j)*m+1  )
%     each g(k) and 0 is a matrix of n x m.  There are NU matrices of
%     0 in the end  of GTEMP.
%
for i=N2:-1:1,
      temp=[];
      for j=1:n,
         temp=[temp; G((j-1)*N2+i,:)];
      end;
      gtemp=[gtemp,temp];
end;
gtemp=[gtemp zeros(n,NU*m)];
%
%     PICK UP PARTS OF MATRIX gtemp AND CREATE MATRIX g, WHICH
%     IS VECTORIZED FORM OF THE STEP RESPONSE MATRIX FOR n-OUTPUT
%     m-INPUT SYSTEM.
%
g=[];
for i=N1:N2,
      temp2=gtemp(:,(N2-i)*m+1:(N2-i+NU)*m);   % NU varies from 0 to NU-1;
      g=[g;temp2];                              %i.e J has variation of u(t+0) to u(t+NU-1)
end;
%
% Q is a diagonoal matrix of m*NU x m*NU (DEFINED IN INIT_MV).
%     Q=diag(Q(0),Q(1)...Q(NU-1))
g=[g;Q];
%     The overall dimension of g is n(N2-N1+1)+m(NU)  x  m(NU);
return
```

Function: mgenf.m

```
function [f]=mgenf(A,B,yf,duf,T,m,n,p,q,maxdel,td,N2,TN2,f)
%
%      [f]=MGENF(A,B,yf,duf,m,n,p,q,maxdel,td,N2,TN2,f)
%      THIS FUNCTION CALCUATES THE FREE RESPONSE OF THE PROCESS.
%
%      This function calculates the value of output for zero input
%      for the horizon of   1  to   N2  in the matrix f;  It is done by
%      calling itself repeatitively and evaluating system output for
%      zero input.
%
%      A      matrix having diagonoal elements of Ay = B del(U), in (q-1)
%      B      matrix of Ay = B del(U) in terms of (q-1)
%      yf      previous filtered plants output from yf(t-1)
%                to yf(t-p);  matrix order  (n x p)
%      duf     previous filtered controller outputs from u(t-1)
%                 to duf(t-maxdelay-q);   matrix order (m x (p+maxdel))
%      maxdel maximum delay of the system (scalar)
%      td     delay matrix (n x m)
%      T      T-filter
%      p      order of numerator polynomials in A+1;
%                  for (a0 +a1q-1 +a2q-2), p=3
%      q      same as p, but for matrix B
%      N2,TN2 maximum costing horizons.
%
%   Rajendra Mutha;  July 4, 90.

sizeduf=size(duf);

%     FOR CALCUATING RESPONSE OF PROCESS FOR 'du'= 0 ,
%     UPDATE MATRIX duf WITH A COLUMN OF ZEROS AND FILTER
%     IT TO GENERATE duf.
duf=[zeros(m,1) duf(:,1:sizeduf(2)-1)];
duf=filt_t(T,duf);

%     CALCULATE THE PROCESS OUTPUT RESPONSE FOR THE INPUT du = 0.
%     NOTE THAT THIS RESPONSE IS THE RESPONSE yf AND NOT y.
temp=mcaly(A,B,yf,duf,m,n,p,q,maxdel,td);

%     UPDATE THE OLD OUTPUT RESPONSE MATRIX WITH THE CALCUATLATED RESPONSE.
sizey=size(yf);
yf=[temp yf(:,1:sizey(2)-1)];

%     DECONVOLUTE THE T-filter TO CALCULATE THE y FROM yf AND STORE IN
%     FREE RESPONSE MATRIX f.
f(:,N2-TN2+1)=defilt(T,yf);

%     DECREMENT THE NUMBER OF TIMES THIS PROCESS HAS TO BE ITERATED AND
%     REPEAT THE ABOVE STEPS TILL THE DECREMENTED VALUE IS NON-ZERO.
TN2=TN2-1;
if TN2~=0,
      f=mgenf(A,B,yf,duf,T,m,n,p,q,maxdel,td,N2,TN2,f);
end;
return;
```

Function: genF.m

```
function [F,flag]=genF(setpt,f,i,N1,N2,NU,m,flag,step_size)
%
%     [F,flag]=genF(setpt,f,i,N1,N2,flag)
%     This m_file will generate the F as required by the
%     m_file of lsi for minimizing the costfunction ||Ex-F||^2;
%
%   Rajendra Mutha;  July 4, 90.

%     VECTORIZE THE SETPOINT, w,  FOR APPROPRIATE N1,N2;
%     FORM THE FREE RESPONSE VECTOR F FROM THE MATRIX F AND VECTORIZED
%     SETPOINT.  APPEND ZEROS AT THE BOTTOM OF VECTOR F TO ACCOUNT FOR
%     THE DIMENSIONAL CONSISTENCY WITH THE MATRIX E = [ G +(lambda)*I].
%     THIS IS DUE TO THE PRESENCE OF (lambda)*I PRESENT IN E.
if (rem(i,step_size)==0),
      flag=~flag;
end
      w=setpt(:,flag+1); setpoint=w'
tempw=[];tempf=[];
for k=N1:N2,
     tempw=[w;tempw];
     tempf=[tempf;f(:,k)];
end;
F=tempw-tempf;
F=F(:)';F=[F,zeros(1,m*NU)]';
return
```

Function: genlim.m

```
function[lim]=genlim(L,U,DL,DU,uu,NU);

%     [lim]=GENLIM(L,U,DL,DU,uu,NU)
%     The function GENLIM generates the vector lim, which has the
%     constraint  of equation (CONSTRAINT COEFF. MATRX)*du>=lim;
%     This is as per the requirements of function 'lsi.m'
%
%     Rajendra Mutha;  July 4, 90.


%     VECTORIZE THE CONSTRAINTS AND APPEND THEM IN THE SAME ORDER AS
%     THE CONSTRAINT MATRIX conmat.
%
tempL=[];tempU=[];tempDL=[];tempDU=[];
for i=1:NU,
      tempL=[tempL;L];
      tempU=[tempU;U];
      tempDL=[tempDL;DL];
      tempDU=[tempDU;DU];
end;

L=tempL-uu;U=tempU-uu;DL=tempDL;DU=tempDU;
lim=[L;-U;DL;-DU];
return;
```

Function: filt_t.m

```
function[yf]=filt_t(T,yf)
%
%     [yf]=filt_t(T,yf);
%     This algorithm filters the first column of y
%     with the T polynomial.
%
%     y  matrix of process output with y(1,:) unfiltered
%     T  vector of T filter.


lentt = length(T);  sizy  = size(yf);
if (lentt==0 | lentt > sizy(2) )
      error('filter size bigger  y in filt.m or T is null')
end
T(:)';
if (lentt ~= 1)
      for i=1:sizy(1)
            temp =0;
            for j=2:lentt
               temp= temp  - T(j)*yf(i,j);
            end
            yf(i,1)=(yf(i,1) + temp);
      end % for i
end % if
return
```

Function: defilt.m

```
function [y]=defilt(T,yf)
%
%     [y] = defilt(T,yf)
%
%     THIS FUNCTION DECONVOLUTES (DEFILTERS) THE T-FILTER FROM THE FIRST
%     COLUMN OF A GIVEN MATRIX AND RETURNS IT AS A COLUMN VECTOR.
%
%     Rajendra Mutha; July 4, 1990.
%
lentt=length(T);
sizyf=size(yf);
y=zeros(sizyf(1),1);
for i=1:lentt
  y = y + yf(:,i)*T(i);
end
return
```

Main Program Script File: mimo_gpc.m

```
%     SCRIPT FILT::: mimo_gpc.m
%
%
%     THIS IS THE MAIN FILE OF THE ALGORITHM mimo_gpc.  THE PROBLEM
%     TO BE SIMULATED MUST BE DEFINED IN THE SCRIPT FILE 'init_mv.m'.
%     init_mv.m EXPLAINS VARIOUS VARIABLES USED IN THIS ALGORITHM.
%
%     Rajendra Mutha;  July 4, 90.
```

```
init_mv
uu=[zeros(NU*m,1)];  % SET INITIAL VALUES OF THE ABSOLUTE CONTROL SIGNAL TO 0.


      % displacement vector output from time 0 i.e ignore uu(:,0);
      % y starts from time 1 with y(1)=[0];
      % In iter i=1, u(1)  and corresponding y(2) are evaluated.
      %
      % AT ITER =i:: u(i-1) AND y(i) AVAILABLE;  CALCULATE u(i);
      % IMPLEMENT u(i) AND EVALUATE y(I+1);

flag=0;sizedu=size(du);sizey=size(y);
yf=y;duf=du;
clc


%     CALCULATE THE E-MATRIX AS REQUIRED BY THE FUNCTION 'lsi' FROM G-MATRIX.
E=mgeng(G,N1,N2,NU,Q,n,m);

for i=1:ITER ,
home
i
%     GENERATE THE FREE RESPONSE OF THE PROCESS.
f=mgenf(A,B,yf,duf,T,m,n,p,q,maxdel,td,N2,N2,f);
[F,flag]=genF(setpt,f,i,N1,N2,NU,m,flag,step_size);


%     GENERATE THE UPPER AND LOWER BOUNDS ON THE MANIPULATED VARIABLES
lim=genlim(L,U,DL,DU,uu(:,i),NU);    %uu--displacement vector of output;

%     MINIMIZE THE CONSTRAINED COST-FUNCTION TO CALCULATE THE CONTROL SIGNAL
      [tempdu,chek]=lsi(E,F,conmat,lim);
%tempdu=pinv(E)*F; %USE THIS LINE FOR UNCONSTRAINED CALCULATION INSTEAD OF LSI.


In(:,i)=tempdu;   % 'In' STORES THE du SIGNALS FOR THE COMPLETE SIMULATION.

%     CALCULATE THE ABSOLUTE VALUE OF THE SIGNAL FROM du (DELTA u)  SIGNAL.
uu(:,i+1)=uu(:,i)+In(:,i);

if (chek==1),error('check for incompatible constraints'),end;

%     UPDATE THE CALCULATED du MATRIX WITH CURRENT CALCULATED CONTROL SIGNAL.
du=[In(1:m,i), du(:,1:sizedu(2)-1)];

%     CALCULATE PROCESS OUTPUT FOR THE IMPLEMENTED SIGNAL
out(:,i+1)=mcaly(A,B,y,du,m,n,p,q,maxdel,td);

%     UPDATE THE CALCULATED y MATRIX WITH CURRENT CALCULATED OUTPUTS.
y=[out(:,i+1), y(:,1:sizey(2)-1)];

%     THE BELOW IF STATEMENTS ARE JUST FOR NEAT DISPLAY OF CONTROL AND
%     CONTROLLED SIGNALS DURING SIMULATION.
 if ( i < 4)
    control_vector=uu(1:m,i:i+1)
    system_output=out(1:n,i:i+1)
else
    control_vector=uu(1:m,i-3:i+1)
    system_output=out(1:n,i-3:i+1)
end

%     UPDATE THE CONTROL AND CONTROLLED SIGNALS AND FILTER THEM THROUGH T.
duf=[in(1:m,i), duf(:,1:sizedu(2)-1)];
yf=[out(:,i+1), yf(:,1:sizey(2)-1)];
duf=filt_t(T,duf);
yf =filt_t(T,yf);

end;   % REPEAT THE ITER LOOP
```